NASA/TM-2006-214552

Diagnostic Technology Evaluation Report For On-Board Crew Launch Vehicle

Sandra Hayden, Nikunj Oza, Robert Mah, Ryan Mackey, Sriram Narasimhan, Gabor Karsai, Scott Poll, Somnath Deb, Mark Shirley

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California, 94035-1000

September 2006

NASA/TM-2006-214552

Diagnostic Technology Evaluation Report For On-Board Crew Launch Vehicle

Sandra Hayden
QSS Group, Inc.
Ames Research Center, Moffett Field, California

Nikunj Oza
Robert Mah
Scott Poll
Mark Shirley
Ames Research Center, Moffett Field, California

Ryan Mackey
Jet Propulsion Laboratory, Pasadena, California

Sriram Narasimhan
University Affiliated Research Center
Ames Research Center, Moffett Field, California

Gabor Karsai
Institute for Software-Integrated Systems
Vanderbilt University

Somnath Deb
Qualtech Systems Inc (QSI)

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California, 94035-1000

September 2006

Available from:

NASA Center for AeroSpace Information
7121 Standard Drive
Hanover, MD 21076-1320
(301) 621-0390

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
(703) 487-4650

National Aeronautics and
Space Administration

Intelligent Systems Division
NASA Ames Research Center

DIAGNOSTIC TECHNOLOGY EVALUATION REPORT FOR ON-BOARD CREW LAUNCH VEHICLE

July 28th, 2006

Sandra Hayden, *QSS Group, Inc., NASA Ames Research Center*
Nikunj Oza, *NASA Ames Research Center*
Robert Mah, *NASA Ames Research Center*
Ryan Mackey, *NASA Jet Propulsion Laboratory*
Sriram Narasimhan, *UARC, NASA Ames Research Center*
Gabor Karsai, *Institute for Software-Integrated Systems, Vanderbilt University*
Scott Poll, *NASA Ames Research Center*
Somnath Deb, *Qualtech Systems Inc (QSI)*
Mark Shirley, *NASA Ames Research Center*

TABLE OF CONTENTS

1 Introduction

1.1 Purpose

To evaluate the state-of-the-practice in embedded fault detection and diagnosis technologies, for requirements development for the Crew Launch Vehicle (CLV).

1.2 Scope

In the decades since Apollo and the development of Space Shuttle, many new and diverse diagnostic technologies have been developed that present opportunities for improved fault detection and diagnosis. The Apollo Emergency Detection System (EDS) monitored just a handful of parameters against redlines and triggered an abort if these limits were exceeded. (Abort decision checkpoints are listed in Table 8 and their usage shown in Figure 2 of Appendix A.)

It is possible to monitor many more parameters on-board Space Shuttle, by using system databuses for vehicle data acquisition and distribution. Space Shuttle Caution and Warning (C&W) utilizes programmable logic to monitor each input signal against allowable limits, which are set manually using thumbwheels. The dedicated C&W system is backed up by Fault Detection and Annunciation (FDA) software in the System Management General Purpose Computer (GPC). FDA compares the sensed values of selected measurements against preset upper and lower limits. If limit boundaries are exceeded, depending on the urgency, action is taken to either reconfigure to an alternate path or annunciate the situation to the crew for appropriate recovery.

This report evaluates current diagnostic technologies for fault detection and diagnosis on-board CLV. The goal of the report is not to recommend technologies to be hosted on-board CLV; rather, it is to identify what is state of the practice that could realistically be flown and what is leading-edge state of the art that cannot be ready for flight, then to use this knowledge to assist the development of feasible requirements for CLV fault detection and diagnosis.

Many diagnostic technologies—both diagnostic algorithms and commercial-off-the-shelf (COTS) products—were initially considered, as described in Section 1.2. An overview of a subset of these technologies that are potentially suitable is provided in Section 2, assessment criteria are defined and an evaluation methodology is laid out in Section 3, and detailed evaluations of these diagnostic technologies based on the assessment criteria are tabled in Section 4. Required fault coverage for crew escape is elaborated in [i] and potential avionics architectures for deployment in [ii]. Assessment criteria developed in this report may be used to recommend requirements for CLV fault detection and diagnosis that are feasible for the current state of the practice [iii].

1.3 Technologies Considered

Many technologies were investigated that did not make the cut for further evaluation. The Honeywell survey of diagnostic tools for the Space Launch Initiative in 2003 [1] and the Ames survey of diagnostic tools performed for the Second Generation Reusable Launch Vehicle Program in 2002 [2] were examined for applicable technologies, and Ames subject matter experts were consulted on state-of-the-art diagnostic technologies. Technologies that were considered and set aside for possible later evaluation as ground operations and maintenance technologies and for design and testability analysis are listed in Table 1. Technologies are identified as COTS products or algorithms, and the vendor or developer is listed. The rationale for considering a technology suitable for ground applications but not for on-board CLV is briefly mentioned. Promising technologies that are relevant to on-board fault detection and diagnosis are listed in Table 2. This is the set of technologies that are further evaluated in the remainder of this report.

Drivers for the allocation of the technology to on-board/ground systems include the need for abort fault detection to be performed on board for fastest possible crew escape; the need for confirmation or certainty in the abort fault detection; and characteristics of the type of faults detected by the technology (e.g., criticality, time to criticality (TTC), mission phase, physics of the fault, and subsystems involved in the fault). Non-critical faults have lower priority than abort faults and their fault detection and isolation may be supported as resources permit.

Data-driven methods such as machine learning and time-series algorithms can perform pattern recognition after being trained on empirical data. These algorithms are not adaptive and do not change once trained. Unsupervised learning algorithms are not suited for abort fault detection; since these algorithms are trained on only nominal data, all anomalies are flagged without discrimination between critical and non-critical faults. Supervised learning algorithms train on labeled nominal and fault data and are capable of identifying critical faults, making these algorithms feasible for abort fault detection. In general, data-driven approaches are not sufficiently mature for on-board application; however, exceptional applications with a history of maturation at NASA are evaluated in this report.

1.4 Applicable Documents

i. CLV Trade Study 2.8.2.8-04 (*aka* AT-0004), NASA Marshall Space Flight Center (MSFC)/NASA Ames Research Center.
ii. CLV Trade Study 2.8.2.8-05 (*aka* AT-0005), NASA MSFC/Ames.
iii. CLV Fault Detection, Diagnosis and Recovery (FDDR) Requirements, NASA MSFC/Ames.
iv. NASA's Exploration Systems Architecture Study (ESAS) Final Report, NASA-TM-2005-214062, November 2005.

Table 1: Ground Operations, Maintenance, and Design Technologies

Technology	Provider	Application	Rationale Discouraging On-board Use
TEAMS	Qualtech Systems Inc (QSI) http://www.teamqsi.com	Design/testability analysis	Design tool.
eXpress	DSI International http://www.dsiintl.com	Design/testability analysis	Design tool.
TEAMS-RDS, TEAMATE	Qualtech Systems Inc (QSI) http://www.teamqsi.com	Maintenance	Maintenance tool.
TestBase	TYX http://www.tyx.com	Maintenance	Maintenance tool.
Maintenix	Mxi Technologies http://www.mxi.com	Maintenance	Maintenance tool for aviation.
Case-based diagnosis	Algorithm	Maintenance	Maintenance tool. Needs in-the-field training to develop the case base.
I-Trend	Scientific Monitoring Inc. (SMI) http://www.scientificmonitoring.com	Condition-based maintenance	Ground-based and avionics applications, e.g., trend monitoring for turbine engines.
G2 Optegrity	Gensym http://gensym.com	Process industries, real-time mission-critical operations monitoring	Support problem: Gensym has a large customer base. NASA needs priority for on-board software providers.
QMC Suite of Programs	Quality Monitoring and Control (QMC) http://www.qmc.net	Process industries	Engineer's desktop software. Not intended for real-time embedded applications.
CBMi, PHM	Impact Technologies LLC http://www.impact-tek.com	Prognostics, remaining life	No hard real-time fault detection. Case studies are faults that develop over hours/days.
RODON	Sörman Information & Media AB http://www.sorman.com	Quantitative/continuous model-based diagnosis	Support problem: vendor based in Sweden.
SensorMiner	Interface and Control Systems (ICS) http://www.interfacecontrol.com	Rule induction and real-time rule-based expert system. (Learns rules from data sets for real-time anomaly detection.)	Low technical readiness level (TRL), few applications.
Fuzzy Logic Intelligent Diagnostic System (FLIDS)	GeoControl http://www.geocontrol.com	Combines fuzzy logic, expert system, and neural network for diagnosis	Low-TRL Small Business Innovation Research (SBIR) technology. Novelty too high. Runs only on user input (no models/code of the system).

Data Analysis System (DAS)	EDO Electronic Systems Group (formerly AIL Systems Inc.) http://www.edocorp.com/	Pre-flight test equipment, post-test data analysis, ruggedized computers	No significant diagnostics capability. Not intended for real-time embedded applications—runs on VAX computers.
ILOG Rules	ILOG Inc http://ilog.com/	Rule-based system	Off-line support. Current business is business rule management systems.
alert	COGSYS Ltd. http://www.cogsys.co.uk	Monitoring and diagnosis of rotating machinery	Support problem: vendor based in the UK.
TIBCO Hawk	Talarian (now TIBCO) http://www.talarian.com	Network monitoring	Tool for system administrators.
ClickFix	ClickSoftware http://www.clicksoftware.com	Troubleshooting	Off-line support. Current business is call center diagnostics for customer problems.
SHINE	Algorithm NASA JPL	Rule/model-based diagnosis system	Low TRL.
MEXEC	Algorithm NASA JPL	Real-time diagnosis with compiled models	Low TRL.
Mini-ME	Algorithm MIT	Real-time diagnosis with compiled models	Low TRL.
TITAN	Algorithm MIT	Reactive diagnosis	Low TRL.
Ace	Algorithm UCLA	Model compilation for real-time diagnosis	Low TRL. Little practical work, mostly theory.
Livingstone (L2)	Algorithm NASA Ames http://opensource.arc.nasa.gov	Qualitative/discrete model-based monitoring and diagnosis	Model-building not easy. No flight demonstration of hard real-time performance.
Hybrid Diagnostic Engine (HyDE)	Algorithm NASA Ames	Discrete continuous model-based monitoring and diagnosis	Low TRL.
Temporal Causal Graphs (TCG)	Algorithm Vanderbilt University	Bond graphs for fault modeling	Low TRL.
IMS and ORCA	Algorithm NASA Ames	Unsupervised learning by clustering	Low TRL. Anomaly detection has potential for false positives.
Probabilistic/ Possibilistic Reasoning	Class of algorithms	Bayesian Nets, Dempster-Shafer, Fuzzy Logic, Markov Chain Monte Carlo	Non-deterministic if probabilistic inference is based on samples drawn from probability distributions. Deterministic with *a priori* probabilities.

Machine Learning	Class of algorithms (supervised learning of labeled data; unsupervised learning of nominal data)	Neural networks, Decision trees, Classifiers and ensembles, Clustering (Kernel/K-means) Support Vector Machines, Principle Component Analysis	Low TRL. Anomaly detection can cause false positives, e.g., unsupervised learning. Real-time application and transparency for verification and validation (V&V) and substantiation of diagnosis can be challenging.
Time Series	Class of algorithms (predicts statistical properties of data sequence and detects anomalies)	Kalman Filtering, Boundary Modeling, Change-point detection, Auto-Regressive Moving Average (ARMA) models, Hidden Markov Models, Dynamic Probabilistic Networks (DPNs), Entropy-based anomaly detection	Low TRL. Anomaly detection can cause false positives.

Table 2: Technologies Applicable to On-Board CLV

Technology	Provider	Algorithm	Application/ Critical Need	Rationale for On-board Relevance
Thruster Fault Detection and Mass-Property Identification	NASA Ames	Maximum likelihood, Recursive least squares	Attitude Control Guidance, navigation, and control (GN&C)	Proven at NASA, real time. Meets critical fault detection needs of GN&C.
TFPG	ISIS at Vanderbilt U	Model-based diagnosis	Propulsion/propagating faults/system-level health management	Approach meets needs for abort fault detection. Also supports system-level fusion.
DIAD (BEAM)	NASA JPL	Time series: Auto-Regressive Moving Average (ARMA) models	Propulsion Vibration Monitoring	Proven at NASA, real time.
SIE (BEAM)	NASA JPL	Statistical covariance	Sensor Validation	Proven at NASA, real time.
SureSense	Expert Microsystems	Statistics and machine learning: Multivariate state estimation, Bayesian probability methods	Sensor Validation	Proven at NASA, real time.
TEAMS-RT	Qualtech Systems Inc	Model-based diagnosis	System-level health management	Proven in industry, real time. Toolset supports full life cycle.

2 Overview of Diagnostic Technologies

On-board fault detection and diagnosis must address the critical functions of the CLV—propulsion and attitude control—in real time. In this section, algorithms from control theory, fault modeling approaches, sensor validation algorithms, and industry solutions are evaluated for embedded abort fault detection and integrated health management. For each technology, CLV on-board operations needs that are addressed are identified, current solutions in use are described, and benefits of the technology are discussed. The methods involved in the technology are presented at a high level and applications fielded are referenced.

2.1 Fault Detection for Attitude Control

The "pencil" configuration of the CLV/CEV stack has been identified as a risk for stability and attitude control. The algorithms in the following sections address the heightened need for situational awareness of vehicle status for attitude control and robust fault tolerant control.

2.1.1 Real-Time Fault Detection for Thrusters

Reaction Control System (RCS) thrusters are critical for CLV roll control during First Stage ascent and pitch control during First Stage separation, for Upper Stage attitude control and separation, and possibly for active structural dampening. For CEV, the RCS thrusters are critical during Upper Stage separation, on-orbit maneuvers, and reentry as well as autonomous rendezvous and docking (AR&D) operations.

Representing the current state of the practice, the Space Shuttle Orbiter's RCS comprises 44 thrusters and multiple fuel and oxidizer tanks, manifolds, fail-operational /fail-safe solenoid valve pairs, drivers, and electrical power. This makes for an extremely complicated redundancy management scheme, for which real-time thruster fault detection and identification (FDI) is essential. Possible Orbiter thruster faults include: 1) incorrect thruster commands from one of the four redundant GPCs' reaction jet driver units, 2) misfiring thrusters, 3) failed-on thrusters, 4) leaking thrusters, and 5) failed-off thrusters. Detecting the status of the thrusters, identifying thruster faults, and isolating thruster failure modes require comprehensive, timed checking of thruster firing commands with thruster manifold pressure, thruster temperature, and navigation sensors.

Recent advances have significantly improved the speed and accuracy for determination of thruster status [3]. In 1976, MIT/Draper Lab developed a maximum-likelihood method for detecting leaking thrusters for the Space Shuttle orbiter's RCS jets. This maximum-likelihood method for detecting soft failures has been extended to detect hard RCS jet failures (failed-on, failed-off), and determine thruster strengths based solely on information from vehicle gyro sensors. Real-time thruster strength information allows for more precise attitude control.

The algorithm for thruster FDI is capable of reliably detecting and identifying hard, abrupt single- and multiple-jet on- or off-failures based on vehicle motion as measured by gyro response. Detection of an RCS failure occurs within one second and identification of which thruster has failed, and the fault mode, occurs within five seconds. The algorithm can use rate gyro signals only; however, the addition of accelerometer signals improves discrimination between similar failures. It is also possible to utilize the gyros in the Inertial Measurement Unit (IMU). What is important is the gyro noise characteristics—low noise makes for better thruster FDI. The algorithm is computationally efficient and scales better than linearly with the number of failure modes to be identified.

During the ascent phases, thrusters are used for roll, pitch, and attitude control. Instead of requiring many redundant thruster systems to handle thruster failures, as for Shuttle where there are multiple redundancies in some thruster directions, real-time reconfiguration of thrusters could be used for fault tolerance. If a thruster problem occurs during First Stage burn, Upper Stage thrusters, and possibly Crew Exploration Vehicle (CEV) thrusters could be used to compensate. The thrusters would need to be sized appropriately; however, by reducing the required redundancy of thruster systems there will be significant savings in launch mass, hardware complexity, and cost. In the event of an impending launch abort situation, the use of Upper Stage/CEV thrusters (including the redundant thrusters) to regain/maintain vehicle flight control for as long as possible could enable a more favorable selection of abort scenarios and provide additional time for crew escape and survival. In orbit, the CEV would be robust to thruster failures as well. If the CEV RCS is minimally actuated, thruster failures will reduce the degree of maneuverability but will not result in loss of controllability.

The thruster FDI algorithms have been applied in MATLAB simulation and hardware testbeds for four space vehicles: the X-38 Crew Return Vehicle and Mini-AERCam (Mini-Autonomous Extravehicular Robotic Camera), both developed at NASA Johnson Space Center (JSC); the NASA Ames Research Center Smart Systems Research Lab air-bearing vehicle (S4); and the MIT SPHERES (Synchronized Position Hold, Engage, Reorient, Experimental Satellites) experimental spacecraft.

Real-time thruster fault detection and identification was successfully demonstrated with this technology onboard the International Space Station (ISS) in May 2006. Astronaut Jeff Williams conducted the tests with SPHERES. The satellite successfully performed several checkout maneuvers which tested all the thrusters and collected IMU data for the thruster FDI and mass-property identification code, using large angle open-loop rotations to test the differences between thruster mixers that convert desired forces/torques into thruster on-times. The first test was successful when Jeff deployed the satellite on a random orientation, after which the satellite rotated to point at a beacon and then held position for more than one minute. Jeff successfully performed all thruster FDI tests—failed-on thruster FDI, failed-off thruster FDI, multiple-thruster FDI, attitude control without FDI as a control test, and closed-loop attitude control with FDI. Live video indicated that all tests ran within expectations, and subsequent data analysis confirmed this. The next tests are planned to include docking maneuvers, translation tests, and mass-property identification.

2.1.2 Mass-Property Identification

Mass-property identification provides accurate information of the vehicle mass, mass center of gravity (c.g.), and moments of inertia. Mass properties change as fuel is expended and the vehicle configuration changes with staging, and hence need to be continually reassessed.

Major benefits of this capability for CLV First Stage and Upper Stage are:

1) Minimization of fuel consumption by thrust vector control (TVC). Optimally, the thrust vector is accurately applied through the vehicle c.g. Accurate determination of the c.g. location improves propulsion performance by maximizing effective thrust and minimizing attitude corrections.
2) Detection of anomalies related to c.g. (e.g., fuel consumption, fuel slosh). If there is an impending problem with solid rocket booster (SRB) combustion or fuel slosh in the Upper Stage feed system, this may be detected by the deviation from expected mass and mass c.g.

For CLV First Stage, the application will require modeling of the aerodynamic loads in order to accurately determine mass and mass c.g. For CLV Upper Stage, the application may be straightforward if the aerodynamic loads are negligible (particularly at high altitude and the trajectory after max Q). For CEV, an additional benefit is improving AR&D performance by virtue of a more accurate mass property model of the vehicle for the GN&C controller. The CEV application is expected to be straightforward.

Real-time mass-property identification has recently become feasible, with an approach based on conventional mathematical methods [4]. The challenge is that the mass properties and thruster properties are coupled in the vehicle's equations of motion and cannot all be solved simultaneously in linear form. The approach is to identify acceleration created by each thruster from the gyro signals, as thruster acceleration reflects both mass and thruster properties and is the real value of interest from a control, estimation, or FDI standpoint. The equations are manipulated into forms that minimize coupling for two sub-problems, for the c.g. and inverse inertia matrix, which are then solved by application of recursive least squares estimation. Properties of the vehicle that are well known, such as the thruster directions and locations in the structural frame, or the rate of fuel mass expulsion, are utilized in the equations. The computationally efficient algorithms reliably and accurately identify mass properties in the presence of several significant noise sources. There are alternative approaches (e.g., Tanygin and Williams' least squares algorithm, Bergmann's Guassian second-order filter); however, these are significantly more complex and computationally intensive.

Real-time thruster fault detection and mass-property identification are complementary algorithms for attitude control fault detection that have been typically executed together. The mass property identification algorithm by itself does not diagnose faults; rather, it identifies spacecraft center-of-mass and inertia properties and accelerometer bias that are used to improve the model accuracy in the thruster fault detection system.

Zero-g flight test results for SPHERES flown on NASA's KC-135A aircraft are reported in [5]. Demonstrations of real-time mass-property identification are scheduled to be conducted with SPHERES onboard the ISS in July/August 2006. The tests include fuel slosh identification, to measure the liquid CO_2 sloshing along the length of the tank, as well as single-thruster firings to validate the algorithm with and without proof mass.

2.2 Fault Detection for Propulsion

Propulsion system abort faults may roughly be categorized by rapidity of onset (fast, slow, or medium), and as sensor faults or as multiple faults. Fast faults are characterized by a signal which rapidly exceeds its critical threshold. Slow faults develop or propagate gradually from an anomaly. Sensor faults can cause launch delays or could trigger an unnecessary abort. Multiple faults can occur separately or simultaneously, or one fault can trigger a secondary fault in a fatal combination.

CLV needs to detect abort faults for the First Stage, Upper Stage, and Upper Stage Engine (USE) propulsion systems. It is expected that engine health management will be handled by the USE controller, due to tight coupling of engine control and feedback. The USE may use Advanced Health Management System (AHMS) technologies developed for the Space Shuttle Main Engine (SSME)—Real-Time Vibration Monitoring System (RTVMS), Optical Plume Anomaly Detection (OPAD), Linear Engine Model (LEM) and System for Anomaly and Failure Detection (SAFD), developed at MSFC. Rocket engine turbopump machinery operates under the most extreme conditions and failures in the turbopump are generally critical. The state of the practice for turbopump fault detection is the SSME RTVMS, successfully flown on STS-96.

Other propulsion systems that require abort fault detection are the Upper Stage main propulsion system (MPS) (the propellant feed system) and First Stage solid rocket booster (SRB). The Shuttle SRB from ATK-Thiokol will be reused, essentially unchanged. It has only a handful of sensors, mainly pressure sensors in the core. The current state of the practice on Shuttle for fault detection of these systems is monitoring of redline limits.

2.2.1 Timed Failure Propagation Graphs

Timed Failure Propagation Graphs (TFPGs) are a fault modeling approach based on dependency graphs. The fault propagation graphs are built by the engineering experts who understand the failure modes of the system and how failures propagate to visible symptoms. Fault graphs can be built in the early design stages, when system structure and implementation may be uncertain but required functionality and functional failures can be modeled. The TFPG models the progression of a fault over time and the observable effects along the propagation path, including alarms and secondary failures.

Timed failure propagation graphs resemble the fault trees developed by NASA for fault analysis [6], in that discrepancies are traceable to the different faults that can cause them. The difference is that the fault tree has no notion of the progression of a fault over time, nor the sensing and monitoring required to detect the fault. During design, the TFPG has particular value for fault analysis with regard to developing sensing requirements and understanding cumulative timing delays due to propagation of effects across components and subsystems.

At run time, crew and controllers can receive advance warning of the fault from early symptoms, as well as the time remaining for escape, using information in the graph. Confirmation of the abort fault detection is provided as the propagating fault is corroborated by downstream alarms. Action is taken based on the time to criticality, the fault mode, and the mission phase Abort Mode. Alarm notifications are provided to the TFPG algorithm by separate purpose-built monitoring code.

TFPGs are applicable to systems in which faults propagate and have measurable precursors, such as propulsion systems, and that cannot be detected in time by a single redline. The Apollo EDS monitored individual redlines and had very little warning of an abort fault, as the parameters monitored were downstream effects—a catastrophic failure was already underway by the time it could be detected. Abrupt faults are detected by the

TFPG with direct monitoring of the modeled fault, as for redlining; the TFPG has no particular advantage over traditional methods. This technique is most appropriate for incipient faults that propagate or develop, and for intermittent faults. Intermittent faults are handled by the algorithm by maintaining a pair of hypotheses (one for the case when the fault is active, and another one when not) for a limited, engineer-defined time interval. Sensor faults are recognized as false alarms by the TFPG when not corroborated by alarms at related monitoring points and expected time intervals. Failed sensors, once recognized, are not used in the reasoning process. Edges in the graph can be associated with operational modes that enable or disable the connectivity represented by the edge. This allows system mode switching as occurs during different phases of operation, e.g., Abort Modes I-IV or staging. The graph may also be used to track dependencies between faults, such as when a primary fault causes secondary faults in other parts of the system. Where interactions between multiple faults can cause a credible abort fault, the dependencies must be captured. Secondary faults can be handled by the algorithm.

A partial TFPG for the Upper Stage propulsion system is shown in Figure 1. The TFPG is composed of three types of nodes: fault, alarm, and discrepancy nodes. Fault nodes are defined for the fault modes to be detected in the system. Alarm nodes are observations from sensor data. Discrepancy nodes are off-nominal conditions that are the effects of failure modes. Discrepancies may or may not be immediately observable by an alarm, depending upon sensor placement. The graph is formed by connecting edges that link fault nodes to discrepancy nodes or other fault nodes (in the case of cascading faults), and discrepancy nodes to alarm nodes. The edges have temporal constraints, the minimum and maximum times for the propagation of the effect.

In the case of Loss of Thrust, failures manifesting as pressure drop and reduction in propellant flow can be detected along the propagation path. Such failures include tank leakage or underpressure, tank outlet problems (pump failure, ullage ingestion, clogging), engine inlet valve malfunction, feed pipe leakage or clogging, Main Fuel Valve (MFV) malfunction, High Pressure Fuel Turbopump (HPFTP) failure, and low Main Combustion Chamber (MCC) pressure and temperature. Since the Upper Stage is self-pressurizing, the feedback loop models failure of the engine to pressurize the feed system.

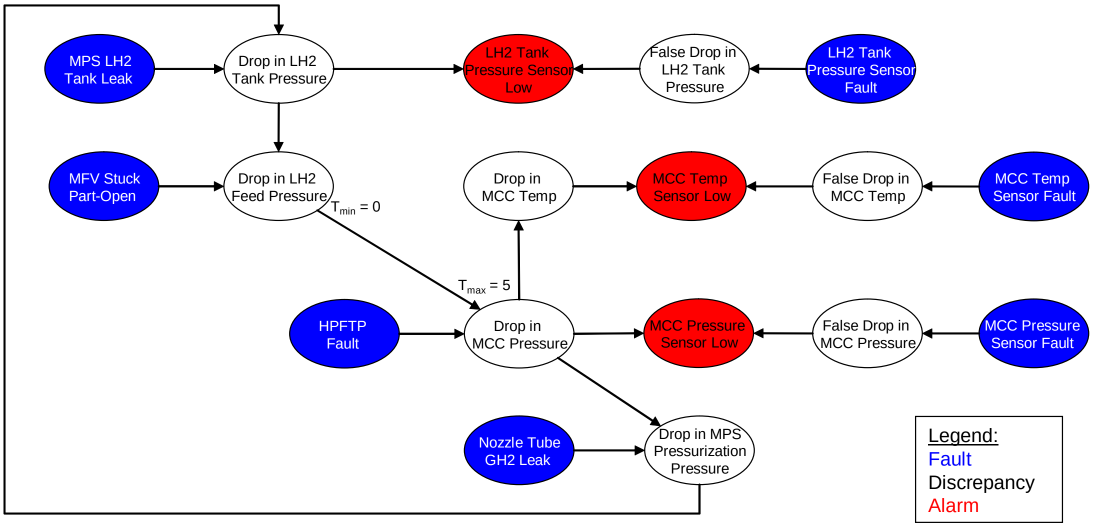

Figure 1: TFPG for Loss of Thrust

The TFPG can also model structure, recommended for organizing larger models. Components are contained in a hierarchy of subsystems and systems and failure modes and discrepancies assigned to components, allowing faulty components to be isolated based on detected discrepancies.

The TFPG algorithm is as follows [7, 8]. On notification of an alarm, the algorithm traverses the graph backwards from the alarm node to identify all fault modes that could have caused the alarm, and time intervals for these faults. Then for each candidate fault, forward traversal identifies future alarms to be expected as this fault propagates, and the time intervals when these alarms should occur from the timing information in the graph. Confidence in the fault diagnosis increases if the alarms occur within the time intervals as expected. Confidence in the diagnosis decreases if the predicted alarms do not occur or do not occur within predicted time intervals, or if unexpected alarms occur. On each iteration, the algorithm regenerates the set of fault candidates to eliminate those that have been ruled out. For a false positive, a transient alarm is not corroborated by other alarms and all fault candidates are eliminated. The algorithm handles system mode changes by dynamic failure propagation, enabling or disabling specific propagation links based on operational mode, and also handles circular dependencies in the graph for feedback loops. Non-monotonic alarms for intermittent faults that become inactive are handled by retraction/recomputation of the hypothesis set, through backtracking and forward propagation. Probability is not explicitly represented in the TFPG; however, the confidence of the diagnosis is based on prior failure modes and the number of substantiated, false or missed alarms. It is possible to guide the search and rank hypotheses based on probability measures such as sensor reliability and the relative probabilities of failure modes, with minor modifications to the algorithm.

The algorithm always attempts to generate multiple hypotheses that explain all the currently active alarms, such that the causal and temporal constraints implied by the graph model are satisfied. The algorithm is biased towards generating a hypothesis that explains an active alarm rather than considering that alarm a “false” alarm. Even if alarms are arriving in an incorrect sequence, the algorithm will generate and update hypotheses that explain those alarms by failure modes, although the confidence and rank of the hypotheses will be low. For an unforeseen fault that was not modeled, there may be no hypotheses that explain the active alarms because the structural and temporal constraints of the model may not be satisfied. In this situation, the generated hypotheses will have low confidence and rank.

The algorithmic efficiency is similar to a graph traversal algorithm. The number of nodes in the graph, *n,* is equal to the total number of failure modes + the total number of discrepancies + total number of alarms. The number of edges, *m*, is equal to the number of propagation dependencies in the system. This gives linear complexity of O ($n + m$). Diagnostic latency is typically sub-second on medium-sized (~400 components) models. However, the usage of AND connectors in the graph (multiple failures cause one discrepancy) needs to be limited, as this causes the complexity to become exponential; in the worst case, all the connected fault propagation paths must be evaluated. The accuracy and correctness of the algorithm has been shown in [9].

The current implementation of TFPG is soft real time, not hard real time; it guarantees second-range response time but not millisecond-range response time. Response time is bounded by the size and complexity of the graph. Worst-case run time and worst-case memory usage can be estimated from the graph and validated by testing, since the algorithm is deterministic. Second-range response time was demonstrated on a Boeing aircraft fuel system and tested with 215 fault scenarios. Techniques that could be explored to improve performance of the algorithm include offline compilation of the TFPG graph into fast lookup tables, partial-evaluation techniques from compiler theory, and refactoring of inefficient code. Earlier experimentation used heuristics for faster performance, however heuristics customize the code for the system and reduce reusability.

The latest implementation of the TFPG diagnostic technology was developed by the Institute for Software Integrated Systems (ISIS) at Vanderbilt University as part of the Fault Adaptive Control Technology (FACT) project. FACT is a project aimed at building a tool suite for constructing advanced control systems, combining fault diagnosis with reconfigurable control. The TFPG diagnosis approach was first developed in the early 1990s, and tested on a small-scale industrial cogenerator system and chemical process plants, in simulation environments.

In 1993, Boeing utilized TFPGs to diagnose faults from the Shuttle flight STS-57 telemetry downlink data stream in an on-orbit test of an ECLSS component for ISS [10]. Boeing has also utilized the FACT tool suite for diagnosability modeling and analysis of the ISS design. In Boeing's J-UCAS program, TFPG was integrated as an embedded algorithm on a VxWorks/PPC platform and exhaustively tested with simulated alarm indications from Matrix-X–based monitors. Boeing has also tested a robust TFPG algorithm on a laboratory heat exchanger system. On the JSF Health Management project with Boeing [11], TFPG was used for diagnostic fusion—performing system-level diagnosis when faults cascade through many subsystems. Boeing engineers have built TFPG models with about 400 components. These large models are currently used as the test set for verifying TFPG capabilities.

The TFPG algorithm has several engineering support tools, including: a Generic Modeling Environment (GME) for building models that has been used on large models (more than 1000 components); a fault pattern simulator for scenario-based testing of the algorithm and for replaying detected discrepancy sequences; translators for importing models in different formats (including DML, in progress); and the Diagnosability analysis tool (DTool) for TFPG, that has been used earlier on the ISS project [12].

In summary, the TFPG approach is simple, direct, and parsimonious. The TFPG approach naturally supports the detection of the fault, determination of the remaining time to criticality from the connectivity of the model, and confirmation of the abort fault. It is most suited for faults that have longer time to criticality, perhaps with human-in-the-loop, making use of its capability for early warning backed up by confirmation of the fault. The algorithm can be developed and supported in-house. The Vanderbilt Technology Transfer Office can make the TFPG algorithm source code available to NASA through an agreement that excludes commercial use. The FACT tool suite that is used to develop the TFPG graph is copyrighted by Vanderbilt University, and supported by ISIS (Institute for Software-Integrated Systems) at Vanderbilt, a university-affiliated non-profit research institute. It is recommended that the TFPG capability be considered in the formulation of feasible CLV abort fault detection and diagnosis requirements.

2.2.2 Dynamical Invariant Anomaly Detector

The Dynamical Invariant Anomaly Detector (DIAD) is a data-driven fault detection algorithm that is a component of the Beacon-based Exception Analysis for Multimissions (BEAM) system from JPL [13]. BEAM is "an end-to-end method of data analysis intended for real-time (on-board) or non-real-time anomaly detection and characterization." BEAM has components responsible for filtering the inputs, analyzing them using several algorithms, keeping track of these analyses over a long period of time to look for degradations, performing prognostic assessments, and fusing of results.

DIAD is based on well-established methods in time series analysis, fitting linear auto-regressive models [13]. DIAD checks individual signals by stationarizing them and then fitting an autoregressive (AR) model to overlapping sliding windows of the signal. That is, a model may be fit on samples 1–100, and then another model fit on samples 11–110, etc., up through the end of the series. Confidence intervals on the AR model's coefficients are calculated. AR models are fitted to new data and if any model coefficient falls outside the 99% confidence interval, that data point is flagged as anomalous. DIAD can be run with raw or smoothed signals, or on the residuals of a system model that is intended to predict the signals.

Multiple datasets are required, for both nominal and anomalous operations, to properly train the AR models. In general, the algorithm requires that there be little variability in the sensor data from normal operation in different tests. High variability in the sensor data will lead to high variability in the AR models' coefficients and large confidence intervals on these coefficients, which in turn leads to an insensitive detector. However, this can be mitigated by applying different AR models under different operating modes and conditions, e.g., the J2 Idle Mode and Full Thrust Mode. Smoothing the data also can mitigate this problem. Depending on the particular application, accurate testbed data may or may not be available prior to flight. For catastrophic failures, there is a need for high-fidelity simulation-based testing or destructive testing if possible. It may also be possible to monitor the engine passively for the first few flights or run as a ground-based algorithm if the telemetry feed is available.

Key benefits of the algorithm are ability to detect unmodeled faults, ability to react to rapid transients, and built-in short-term signal prediction. DIAD analyzes data in the time domain and is especially well suited to detecting shifts or trend changes. This makes DIAD complementary to RTVMS, which analyzes data in the frequency domain and therefore is well suited to analyzing periodic data. DIAD is fast, since it fundamentally is a linear model fitting algorithm, which makes it suitable for real-time use. DIAD can produce false positives if training is poor—for example, only training with calm environment or control inputs, and then running with violent inputs, leads to unexpected transients in the sensor data.

This algorithm has been tested on data from an SSME test firing at Stennis [15], running single-blind on limited training data. DIAD was able to detect major anomalies including HPFTP blade failure, LPFTP and HPFTP cavitation, HPOTP performance shift, fuel flowmeter shift, frozen sense lines, and deactivated sensors. It was unable to detect an anomaly in the HPFTP during a fuel turbine pump cavitation event, because the signal was not modeled well in the training data. The false negative would have been prevented had there been more access to the engine for testing and training data—only a handful of tests were permitted, all with sensor dropouts, with different SSME configurations and showing large variation between engines. Performance improvements can be expected with higher-rate data (the tests utilized low-rate data) and by using partial predict information from SSME models. The SSME linear model simulates all signals for all modes of operation with reasonable accuracy; the specialized power-balance and transient models simulate only certain signals for certain modes, but with high fidelity.

2.3 Sensor Validation

Complex subsystems such as propulsion and GN&C have many sensors. For instance, GN&C may utilize IMUs, gyros, accelerometers, air data sensors, Tactical Air Navigation (TACAN) System, star-trackers, etc. Even with redundancy of control sensors, sensor unreliability remains an issue that needs to be better addressed for CLV. Currently, signal selection for the four Space Shuttle Orbiter rate gyro assemblies (RGA) is performed by selecting the higher of the two mid values. If the input from any unit diverges from the other three beyond a preset threshold, the input is rejected, the rate gyro is declared inoperative, and the mid value of the remaining three inputs is selected. Disqualifying a sensor based on a simple threshold requires the threshold to be set high to avoid false positives under noisy conditions, decreasing the sensitivity of the sensor fault detection mechanism. Certain errors will not be detected, skewing the results of the sensor selection process.

2.3.1 System Invariant Estimator

The System Invariant Estimator (SIE) is a data-driven fault detection algorithm that is a component of the BEAM system. The SIE algorithm is based on well-established covariance matrix methods in statistics. SIE takes as input a pair of signals that have been time correlated (such as by interpolation or decimation) and monitors changes over time in their coherence coefficient, which is the covariance between the signals divided by the maximum of the variances of the two signals. The algorithms operate under the assumption that the coherence between signals decreases when anomalous operation occurs.

There are several ways for a sensor to fail, including sensor drift, shift, spike, freezing, excessive noise, and no signal. The benefit of the algorithm is the ability to detect any sensor failure in which the sensor no longer tracks the readings from other related sensors. This includes detection of freezing, where the sensor reading flatlines, which is undetectable by current thresholding methods when the sensor freezes within the nominal operating range. Even in quiescent conditions where sensor readings are steady, SIE is able to detect in-range faults by the absence of noise on the failed sensor. Sensors do not have to be redundant in order for SIE to track covariance. Sensors that have a fixed relationship can be correlated—for example, pressure and temperature sensors (related by *PV=nRT*) or current and voltage sensors. This also applies to less obvious sensor couplings such as an increase in both vibration and temperature, and does not require that the relationship be known ahead of time.

The complexity of the algorithm is O (N^2T), scaling with the square of the number of signals simultaneously observed N and linearly in time T. Once the number of input signals is determined it is a fixed-cost, guaranteed-time algorithm with no decision points inside the calculation. The algorithm is anytime, in that processing each data sample steadily improves confidence. Since the confidence relates to the number of samples, for any required confidence threshold a corresponding minimum number of samples must be processed. This determines the latency of the detection.

The N^2 scaling does put a practical upper limit on the number of signals that can be processed for a given CPU, but the algorithm is efficient enough that the limit is typically much higher than required and SIE performance has never been found to be a limiting factor. The limiting factors tend to be the demands of accurate modeling and state-space explosion as the number of signals grows large. For instance, a covariance analysis of GN&C sensors would probably require six signals, one for each degree of freedom, with partially redundant instruments or position/attitude estimators for each degree of freedom. With fivefold redundancy, this is just 30 signals. To increase performance, the covariance matrix can be broken into block diagonal elements, for instance considering each degree of freedom independently which reduces N; or based on state information, segment subsequences within the time series and run the algorithm only on those smaller segments, which reduces T.

SIE has demonstrated several successful field tests. SIE has been tested on data from an F-15 aircraft hydraulic system where failures were induced by attenuating the accumulators [15]. By training using data from at least two of three possible operation modes, SIE achieved zero false positives and zero false negatives. Diagnostic latency in the anomaly detection was on average 30 samples or 0.15 seconds, as the dynamics of the failure required several data samples for detection. 90% of abnormal samples were flagged as abnormal; the 10% of data samples not recognized as abnormal all fell into the latency period during which not enough data had been sampled to determine that there was a fault. In this case, latency included controller inputs that were necessary to reveal the faulty behavior. There is a trade-off between latency and false-alarm rate; the latency can be reduced by tolerating an increased false-alarm rate. Depending on the fault signature, it may be possible to use a higher sampling rate to reduce false negatives and diagnostic latency with no increase in false positives.

For Cassini, in 1998 SIE demonstrated covariance-based sensor validation on post-launch flight telemetry [16], correlating their Inertial Reference Units (IRUs) with sun-trackers and star-trackers. This detected a Cassini star-tracker anomaly, related to an invalid software parameter, although physical failures can also be detected. SIE has been demonstrated in a real-time, on-board implementation on the Dryden F/A-18 [17], detecting gradual degradation in the engines systems. The algorithm is also suitable for application to the F/A-18's on-board attitude and navigation system. SIE has also been applied to the X-33 aluminum tank fill test, identifying failures in tank strain gauges [18].

SIE requires training on well-populated datasets for thorough validation, similar to the DIAD component of BEAM. Readiness for flight is dependant on the availability of good data for the particular application. Training on only nominal datasets will flag any off-nominal data as anomalous, which may result in a false positive. Training with simulated or actual failure data with coverage of the full fault matrix allows identification of faults and eliminates false positives. This detects only the faults that have been trained for; unanticipated faults will be missed.

SIE is able to detect subtle differences between related sensors such as degradation, whereas DIAD monitors a single signal and is able to rapidly detect dynamic changes under noisy conditions such as transients, level shift, and drift. On the other hand, SIE will not detect a level shift that is not sampled during the shift, as the shifted reading will continue to track well against other sensors. Only the absolute magnitude of the signal has changed, its relative behavior has not. The two statistical approaches can be used in a complementary fashion to cover both static and dynamic sensor fault detection, and in fact DIAD and SIE have typically been executed together, integrated by the SHINE rule engine, and can train on the same datasets. DIAD and SIE, which both work in the time domain, are complementary to RTVMS, which works in the frequency domain. A weakness of the SSME RTVMS that has been noted is the poor discrimination of the fast Fourier transform between excessive vibration and accelerometer noise. Another approach could be to monitor an engine accelerometer for excessive vibration with DIAD and correlate the vibration data with related temperature data using SIE. An increase in both vibration and temperature could indicate an engine problem with a higher degree of certainty than examination of the vibration data alone, and might improve upon RTVMS results.

2.3.2 SureSense

The SureSense system was developed under a NASA Small Business Technology Transfer (STTR) for Space Shuttle telemetry data monitoring at MSFC. It has been deployed at the U.S. Department of Energy's Pacific Northwest National Laboratory (PNNL) for development of military and energy applications. SureSense combines several statistical and machine learning methods for sensor validation [19]—the Multivariate State Estimation Technique (MSET), the Sequential Probability Ratio Test (SPRT), the Bayesian Sequential Probability (BSP) test, and the Bayesian Conditional Probability (BCP) method. MSET is used to predict a quantity of interest within the system. SPRT and BSP are statistical hypothesis testing methods used to determine whether the prediction is significantly different from the true quantity. BCP analyzes sequences of results from the SPRT and BSP to decide if there is an actual fault present or a false alarm.

MSET uses a kernel regression model to predict the state (or any other variable of choice) as a function of other variables. The residual between the predicted value and the true value is intended to serve as an indication of whether the system is in a normal or abnormal operating state. Kernel regression methods calculate a prediction for new data as a weighted average of the predictions for the training examples (examples for which the predictions are known), where the weights are a function of how close the inputs for the new data are to the training examples. That is, the prediction for the training example that is most similar to the new data is given the highest weight, the prediction for the second most similar training example is given the second highest weight, etc. In kernel regression, the key problem to solve is determining the metric used to measure distance between input data points. MSET uses a proprietary set of nonlinear operators to transform the input data into a different space that is more appropriate for measuring the distance between data points.

Kernel methods are very flexible due to the variety of ways of measuring the distance between data points, and hence the different ways of performing the weighted average over the training data predictions to calculate predictions on test data. One type of kernel method, Support Vector Machines (SVMs), has demonstrated high accuracy, which has caused SVMs to become one of the currently most active areas of research in the area of machine learning. The distance measures used are often nonlinear kernels (e.g., Gaussian, polynomial), which allows kernel methods to capture more complicated nonlinear relationships in the data that linear methods such as SIE do not capture. However, this flexibility comes at the price of being very memory intensive. In particular, the kernel matrix, which represents distance information used in kernel regression, is of size TxT, where T is the number of training data points. In problems where time-series data is collected, T could be the number of time samples or windows in which sensor measurements are recorded, which could be a very large number. The number of training data points used could be reduced by drawing prototype training examples (e.g., in regimes where the measurements are nearly constant); however, this must be done very carefully to avoid the danger of leaving out valuable information. There is also active research on other ways of approximating kernel methods to make them less memory intensive and faster. In tests of SureSense on SSME data [19], tests were limited to selecting only 250 training examples to enable real-time operation. This is since the complexity of testing is the number of tests multiplied by the number of training examples; the complexity of the learning is the square of the number of training examples. However, it was acknowledged that this number of training examples is insufficient to model the system and many false alarms were obtained because of this problem.

SPRT is a statistical hypothesis testing method that is used in SureSense to determine whether MSET's residual is significantly different from zero. SPRT assumes that the residual can be accurately modeled as Gaussian white noise. BSP was developed under the SureSense STTR in order to relax this requirement. In tests, BSP produced fewer false alarms than SPRT because the BSP was able to accommodate the heavier tails of the distributions of the MSET residuals.

The BCP takes a sequence of independent decisions made by SPRT or BSP and decides whether the number of alarms raised is enough to signify an actual fault. It requires five user-specified parameters: the maximum allowed false alarm probability, maximum allowed missed alarm probability, the number of past decisions by SPRT or BSP used to decide if there is a real alarm, the prior probability that there is no alarm, and the confidence level (the threshold such that if the fault probability calculated by BCP is above the threshold, then the system assumes that there is actually a fault). Standard textbook methods of Bayesian inference are used to calculate the fault probability. The one key drawback of BCP is that it requires that at least half the past decisions by SPRT or BSP indicate that a fault is present before it is convinced that a fault is really present. Additionally, it was assumed that each past decision is independent, which is a commonly made, but nevertheless suspect, assumption.

2.4 Integrated Health Management

Complex systems such as CLV span multiple physical domains and engineering disciplines. Subsystem interactions have been recognized as a major source of error and mishap in NASA missions; and with systems evolving into even more complex systems of systems, this trend can be expected to continue. Integrated Health Management (IHM) addresses this gap by monitoring subsystem interactions and fusing diagnostic results into an overall vehicle state of health that provides consistent, coherent, and comprehensive information for situational awareness and situation assessment by crew and controllers. IHM corresponds to the Orient step in the OODA (Observe, Orient, Decide, Act) model of decision support, in which the state of health of subsystems is interpreted in the context of the current mission phase, mode of operation, and known subsystem failures.

In the CLV/CEV programs, NASA will be responsible for system integration of subsystems and components from contractors, and an important system engineering function will be the coordination of health information. It is expected that IHM will have a significant role at NASA in these programs.

2.4.1 TEAMS-RT

TEAMS-RT is a real-time model-based diagnostic engine, a component of the TEAMS toolset, a commercial product developed by Qualtech Systems Inc. (QSI) under multiple Ames SBIR contracts and internal funding [20]. The toolset provides diagnostic modeling, testability analysis, real-time diagnosis, and maintenance support. TEAMS-Designer is the primary design and diagnosability analysis tool for diagnostic models. TEAMS-RDS (Remote Diagnostic Server) and TEAMATE are complementary tools for telemaintenance and field maintenance. The technician runs TEAMATE on a handheld computer or PDA, and accesses TEAMS-RDS via network connection. TEAMS-RDS organizes fleet-wide maintenance and includes the TEAMS-KB database of models and failure modes, repair procedures and on-line documentation in various media (text, photos, videos, schematics, etc), diagnostic test results, and historical data. The technician can conduct interactive diagnostic sessions with procedural guidance from TEAMS-RDS, or browse Integrated Electronic Technical Manuals (IETMs). Managers can generate health status reports for the fleet of vehicles using the TEAMS-KB record-keeping system, and use the information to assess readiness for a mission or schedule required maintenance.

TEAMS-RT is the real-time diagnostic engine suitable for embedded applications. The algorithm is as follows [21, 22]. Initially, the state of all components is Unknown. On each iteration, the algorithm 1) processes passed tests, identifying Good components; 2) processes failed tests, identifying Bad components that are detected and isolated by the tests and Suspect components that may be Bad but are not isolated by specific tests. The tests in TEAMS-RT utilize signal processing or statistical methods on the raw data to produce a “pass” or “fail” test result. The algorithm continuously processes all the test results against the relationships from the TEAMS model.

The production version of TEAMS-RT includes additional capabilities for system mode changes and redundant systems; supporting the update of dependencies between faults and test points in response to system mode changes, and the update of dependencies resulting from failures in redundant components. TEAMS-RT also has the capability to diagnose intermittent faults and to predict hard failures from such intermittent behavior. TEAMS-RT also can reason in the presence of uncertainty, e.g., tests reporting incorrectly or flip-flopping of test results in the presence of noise [23]. TEAMS-RT can indicate the probability and criticality of the failure mode. For Joint Strike Fighter (JSF) Engine Health Management, probability was used extensively to rank the repair actions and this allowed more than 90% accuracy of correct first pull to be achieved [23].

Recent new capabilities include fault propagation delay modeling in TEAMS-Designer, anytime diagnosis under various test reporting latencies, and support for multi-outcome tests and tests with asymmetric confidence on Pass/Fail outcomes [24]. TEAMS-RT does not yet compute TTC from fault propagation delays. In the near future, TEAMS-RT may do real-time impact analysis, which will bring together propagation delays, criticality, redundancy, and real-time fault tree analysis to assess the impact of faults.

TEAMS-RT is a fast and compact algorithm for real-time diagnostics and system health monitoring. The complexity of the algorithm is O (n ln n) where n is the number of nodes in the graph. As for many diagnostic algorithms, there is negligible processing under nominal conditions. Once TEAMS-RT has formed a diagnosis, all the test results can be predicted and there is negligible processing until the next failure event. Real-time performance results for TEAMS-RT [22] for a simulated system with 80 modes of operation, 1000 faults, and 1000 tests are shown in Table 3. The first column is the number of faults injected; T_p is the number of tests passed; next are the number of good, bad, and suspect components; and the last column is the processing time. A single fault look-up takes 50 milliseconds and this increases less than linearly with the number of faults. Similar timings were observed on the X-33 Integrated Propulsion Technology Demonstrator (IPTD) test stand [25].

Table 3: TEAMS-RT Performance Results on a 50-MHz Sparc CPU

Faults	T_p	Good	Bad	Suspect	Time (ms)
1	993	997	1	2	50
2	978	996	2	2	50
5	931	991	5	4	50
10	881	983	10	7	75
20	819	973	20	7	87

TEAMS model development is based on a structural model of the system, with functional and general failures overlaid on this structure. Functional failures propagate by means of signals and a general failure causes failure of all dependent devices. The structural model allows fault isolation to a specific component or LRU. Test points take in test results, and signals model how test failure propagates to the various monitoring points. TEAMS-RT runs a processed version of the TEAMS model that has been converted to a more efficient run-time format called the dependency matrix (or D matrix). Models can also be imported from other tools, e.g., Matlab, PSpice, and other CAD tools. QSI is involved in the development of industry standards for the interchange of diagnostic information, such as IEEE's 1232 Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE) and the emerging Diagnostic Markup Language (DML). AI-ESTATE formats for diagnostic data are supported. TEAMS models can also be imported from other formats, e.g., XML or CSV.

The TEAMS toolset is best suited for condition-based maintenance of fleet vehicles and Integrated Health Management. The state of all modeled components is monitored as Unknown, Good, Bad, or Suspect. This resolution is appropriate for condition-based maintenance, however the information required for abort fault detection is slightly different. Only Bad components can trigger an abort, in order to minimize false positives; and Good or Unknown components are not relevant. For Bad and Suspect components, the time to criticality (TTC) is important information. For Suspect components, the confidence should be reported. TEAMS (and fault modeling approaches in general) are less appropriate for operations monitoring by crew or controllers, as full situation awareness requires tracking of nominal state as well as failure information.

TEAMS-RT has provided advanced health management solutions for numerous aerospace applications, including Pratt & Whitney engines, Sikorsky helicopters (including the UH-60 and SH-60 turbine engines), and the Boeing AH-64D Apache helicopter. Table 4 shows real-time performance for several diagnostic applications as run on a 50-MHz SS20/502 computer. QSI was selected by Pratt & Whitney to supply diagnostic software for the Joint Strike Fighter's F-135 engine, and has validation results from over 12,000 test cases for a recent Air Force engine. NASA applications include the

X-33 IPTD test stand [25], diagnosis for a non-toxic RCS [26], monitoring and diagnosis of ISS telemetry data [27] and QHuMS Health and Usage Monitoring System (HUMS) for the engine and transmission of a UH60A helicopter, with NASA Ames.

Table 4: TEAMS-RT Performance for Various Applications

System	Number of Tests Pass / Fail	Number of Faults inserted / total modeled	TEAMS-RT CPU run time/call
1553	59 / 2	2 / 174	~5 ms
Xmission system	46 / 5	2 / 160	~5 ms
LO2 Feed System	329 / 39	3 / 167	~5 ms
Engine System	274 / 32	3 / 255	~10 ms

Overall, few COTS technologies were considered for on-board fault detection. This is due to the critical nature of the application and the need for complete transparency in V&V of flight code that may conflict with proprietary considerations, as well as the need for the fault detection software to meet NASA-specific requirements that are not yet defined and that may diverge from the product's commercial market. TEAMS, as a suite of complementary tools, has a great deal of capability for full life-cycle support—from testability and reliability analyses for design to on-board ISHM and FDI processing and ground maintenance and turnaround. The toolset meets a broad range of industry requirements and as such may not be a minimal, optimal solution for abort fault detection. The TEAMS tool suite is proprietary, developed and supported by QSI. Full V&V of COTS product source code, such as the on-board TEAMS-RT algorithm and the process which generates the dependency matrix, would need to be negotiated. NASA has recently authorized a $5 million purchase agreement covering diagnostic and maintenance software and services from QSI for the Exploration Systems Mission Directorate (ESMD). In view of these factors, it is recommended that the TEAMS-RT/TEAMS capability be considered in the formulation of feasible CLV integrated health management requirements for non-critical fault detection and diagnosis.

3 Assessment Criteria and Evaluation Methodology

Assessment criteria are defined in terms of Figures of Merit (FOM) and Technical Performance Measures (TPM). FOMs are the major characteristics desired for the diagnostic technology, and are broken down into specific measurable criteria, the TPMs. TPMs that are well known and found to be strong discriminators for the diagnostic technology may subsequently be recommended as requirements.

A technology is evaluated by comparing its performance measurement against a TPM. For instance the technology's fault detection latency is assessed against the Diagnostic Latency TPM, a measure of the Performance FOM. The Diagnostic Latency TPM is determined by the time to criticality for an abort fault, including sufficient margin for crew escape. The technology with diagnostic latency that best meets the required performance is the strongest candidate in terms of this single assessment criterion.

The relative importance of the assessment criteria is indicated by the weight assigned to the TPM, ranging from 1 to 5 with 5 having greatest significance. These weights and numeric assessments could be used by multi-criteria decision analysis methods such as Kepner-Tregoe, Analytic Hierarchy Process (AHP), or Quality Function Deployment (QFD). For this evaluation, the weights are simply used as a guideline for relative importance. TPMs are qualitatively assessed; accurate TPMs require benchmarking an application of the technology. To compare measurements from different technologies, applications would need to have the same diagnostic scope and be tested under identical conditions and fault scenarios. Sensitivity analysis can check the response of an application to varying the weights, using Monte Carlo methods for instance.

3.1 Figures of Merit

The high-level desired attributes of the technology, FOMs, are defined in Table 5. Related Technical Performance Measures (TPMs) are in italics.

Table 5: FOMs and TPMs for CLV Diagnosis

Figure of Merit	Description with TPMs in italics
Coverage	Diagnostic coverage refers to the ability of the diagnostic system to detect *all credible abort faults* and required non-critical faults. *Breadth* of the coverage is the capability to diagnose a variety of problem classes. *Depth* of coverage is the range of fault isolation supported, e.g., from subsystem interactions or functional failures at the system level, to identifying the specific component that has failed.
Performance	Diagnostic performance refers to the ability of the diagnostic system to meet real-time needs, e.g., *latency* for abort fault detection must be less than TTC.
Accuracy	Diagnostic accuracy refers to the *precision*, *confidence,* and *ambiguity* of the diagnoses. On-board, critical faults should be isolated only to the resolution necessary to determine the need to abort and prevent a minor fault from causing an abort. Further fault isolation reduces ambiguity but increases diagnostic latency, and may be done in ground operations.
Integrability	Integrability is the capability of the diagnostic system to *integrate* with and run on the target hardware/software architecture e.g., VMS platform.
Maturity	The diagnostic system should have a track record (e.g., *TRL*, *deployment history*). It should be *reliable* (robust, fail-proof) and *stable* (meets goals and needs).
Scalability	Scalability is the ability of the diagnostic system to *scale up* fault coverage from small to large-scale complex systems, without excessive loss of performance or increase in code size, e.g., O (n log n). In a *distributed architecture*, support for fusion of results from heterogeneous diagnostic systems is needed.
Testability	The diagnostic system should facilitate *full test coverage* through accepted V&V methods. *Requirements traceability* is supported for products of the technology throughout the life cycle. The diagnostic software and test tools should produce *repeatable results*, within real-time requirements. Verification should use a feasible *number of test cases* for the type of input data used by the technology.
Usability	Support for effective *situation assessment* in ground operations. Multi-user interfaces for *collaborative* and interactive environments. *Online documents and data archives* for operations. Support for *knowledge capture and modeling* of the design. *Reusability* of technology products.
Supportability	The diagnostic technology should be *maintainable*. When *upgraded*, the technology should not be brittle or overly sensitive to change. The *supporting organization* should be stable and mature.
Cost	Relative *cost* and *risk* of exceeding available program funds for Design, Development, Test, and Evaluation (DDT&E) of the diagnostic system.
Schedule	Relative *schedule required* and *risk* of exceeding available program schedule for DDT&E of the diagnostic system.
Extensibility	The diagnostic system should extend to *fleet operations* and be *evolvable* for future exploration systems requirements, e.g., Cargo Lunar Vehicle (CaLV) and Lunar missions.

3.2 Technical Performance Measures

Different TPMs are appropriate for the on-board, ground operations, and design aspects of fault detection. TPMs for on-board fault detection are defined in Table 6. TPMs for ground operations fault detection and for design tools that support fault detection are defined in Appendix B, Tables 9 and 10, to show how the deployment environment influences the metrics as well as for possible utilization in later analyses of ground fault detection and design needs.

There is a trade-off between the false positive rate TPM and the false negative rate TPM that must be made based on application requirements. All algorithms have the potential for false positives when analyzing noisy, dynamic systems. Generally the sensitivity of an algorithm can be 'tuned' for the optimal balance. The Receiver Operator Characteristic curve of an algorithm gives false positive/false negative rates as a function of sensitivity.

Required coverage includes credible abort faults that cause Abort Modes 1, 2, 3, and 4 and on-pad abort (criticality I/II faults). On-board abort fault detection is active from the time the CEV Launch Abort System (LAS) is armed on the launch pad until separation of the CEV from the CLV [iii]. Fault coverage for CLV First Stage and Upper Stage will likely be specified in the CLV Failure Modes, Effects, and Criticality Analysis (FMECA), including criticality, symptoms, TTC, and mission phase; required coverage is unknown at this stage. In this evaluation, coverage is assessed by the ability of the technology to detect the types of faults that it is intended for, and the breadth and depth of that fault coverage. Types of faults can be characterized by subsystem or function (e.g., propulsion, electrical, avionics), or attributes such as type of sensor data, dynamics and rapidity of onset of symptoms, and sampling rate required to detect symptoms. Technologies that can detect a wide range of faults will have more general utility; no single diagnostic technology will be able to detect all faults.

Deployment is assumed to be on the Vehicle Management System (VMS), with the VMS platform typical of avionics systems on current NASA spacecraft. This is consistent with current thought on related trade studies. Deployability on firmware is also assessed since it is possible that programmable logic could provide local fault detection for optimal latency or for backup of the VMS in the event of electrical/power faults that disable the VMS fault detection and avionics systems. This would be similar to the Apollo Emergency Detection System FPGA. The ESAS [iv] estimate for CLV fault detection code size is used as a guideline, 10K source lines of code (SLOC). Note that it is not possible to accurately compare real-time performance of different algorithms without benchmarking in a controlled test environment with equivalent models and data pre-processing code.

Table 6: Assessment Criteria for On-Board Fault Detection

FOM	TPM	Weight	Description
Coverage	False Negative Rate = 0	5	100% coverage of required faults. Zero missed abort faults.
	Maximum Breadth Coverage		Range of fault classes covered.
	Maximum Depth Coverage		Multiple levels of abstraction covered.
Performance	Diagnostic Latency	5	Latency < Time to criticality minus margin for crew escape. Maximize time for crew escape, minimize detection delay. Guarantee diagnosis in hard/soft real time.
Accuracy	False Positive Rate = 0	5	Zero false alarms that trigger unnecessary crew abort, including sensor failures.
	Ambiguity		Isolates to level necessary to determine need to abort.
	Probability of diagnosis = 1.0		Confidence in the abort recommendation.
Integrability	Integrates on VMS platform	4	Compatibility with VMS Real-Time Operating System (e.g., VxWorks) and processor (e.g., PowerPC). Diagnostic application encoded in high-level language (e.g., C/C++) with restricted size (e.g., 10K SLOC) and run-time memory.
	Integrates on EDS platform		Compatibility with programmable logic chips, e.g. FPGA.
	Data I/O compatibility		Supports integration via application programming interface (API) for VMS data acquisition (e.g., 1553 data bus) and output of diagnostic results (e.g., CLV/CEV storage devices and telemetry downlink).
Maturity	TRL ≥ 6	4	Current Technology Readiness Level.
	Deployment history		Number of relevant deployments in NASA, aerospace industry.
	Reliability		Robust and fail-proof, based on bug-tracking history and user problem reports.
	Stability		The current version of the technology meets its intended purpose.
Scalability	Proven in large-scale system	3	Relevant deployments in NASA, aerospace industry.
	Scale-up		Supports fault coverage for First Stage, Upper Stage, and Upper Stage Engine.
	Distributed system		Supports fusion of results from First Stage, Upper Stage, and Upper Stage Engine diagnostic systems.

Testability	Full test coverage	4	Conventional verification for all required fault scenarios. 100% flight code coverage. Specialized V&V tools meet reliability and certification requirements.
	Traceability		Requirements traceable through DDT&E artifacts.
	Repeatable tests		Deterministic diagnostic software/V&V tools. Repeatable results are obtained within real-time requirements.
	Number of test cases		Verification of expected results for all input data used by the technology can be verified with a feasible number of test cases, e.g., inputs are discrete and sparse, or continuous and the range can be broken up into discrete sets covering special/ boundary cases and exceptions.
Usability	Reusability	4	Diagnostic system design factors out common reusable parts, e.g., object-oriented software with code reuse, model-based diagnosis with reuse of the diagnostic engine, and reuse of generic component models.
Supportability	Maintainability	4	Effort to maintain the diagnostic system after deployment. Availability of manuals, user guides.
	Upgradability		Effort to upgrade the diagnostic system after deployment. Sensitivity to change, based on deployment history or degree of partitioning/reuse in the design.
	Quality of supporting organization		Support for DDT&E, maintenance or upgrade. Stability and capability maturity model integration (CMMI) level of the organization. Availability of knowledgeable experts throughout the life cycle.
Cost	Relative cost	3	Relative cost for this technology, e.g., labor for DDT&E, licensing, materials
	Highest cost risk		Likely highest cost risk for this technology.
Schedule	Relative schedule	3	Relative schedule for this technology, e.g., time for DDT&E and acquisition.
	Highest schedule risk		Likely highest schedule risk for this technology.
Extensibility	Fleet operations support	2	Use of fault information across the fleet for logistics supply and maintenance. Use of historical data for prognostics and preventative maintenance.
	Evolvable		Applicability to Exploration Systems, e.g., CaLV and Lunar missions.

4 Detailed Evaluations of Diagnostic Technologies

Results of evaluations against the assessment criteria are tabled in this section.

The symbols in the tables are intended to rate the degree of compliance with the TPM, and may be mapped to numerical values for analysis by decision theory methods:

✓	– satisfied.
~	– partially satisfied.
✗	– not satisfied.
?	– satisfaction is to be determined.
!	– alert/pitfall.

Table 7.1: Evaluation of Thruster Fault Detection with Mass-Property Identification

FOM	TPM	Weight	Description
Coverage	False Negative Rate = 0	5	✓ In extended simulation testing on the X-38 vehicle, 99.98% of thruster failures were identified.
	Maximum Breadth Coverage		✓ Applicable to any attitude control subsystem, RCS or TVC. ✓ Thruster algorithm detects and identifies thruster failures that affect spacecraft motion: hard, abrupt, single- and multiple-jet on- and off-failures and leaks. ✓ Mass-property algorithm identifies spacecraft center of mass and inertia properties using gyro signals and can be extended to reaction wheels/control moment gyros (CMG).
	Maximum Depth Coverage		✗ Thruster/mass-property algorithms are not hierarchical and do not have a range of fault isolation.
Performance	Diagnostic Latency	5	✓ Thruster faults are detected within one second, isolated in one to five seconds.
Accuracy	False Positive Rate = 0	5	✓ Center of mass is determined accurate to better than +/- 5 mm, in testing on the JSC Mini-AERCam spacecraft, using only gyros. Accuracy can be improved by also using accelerometer data. ✓ Thruster algorithm will not produce false positives given accurate sensor data. Mass-property algorithm corroborates thruster faults with c.g. to reduce false positives and does not diagnose faults.
	Ambiguity		✓ Extensions to augment gyro signal data with accelerometer signals have been developed and improve discrimination between similar thruster failures.
	Probability of diagnosis = 1.0		✓ Maximum likelihood is associated with the thruster fault diagnosis.
Integrability	Integrates on VMS platform	4	✓ The thruster fault detection and mass-property algorithms have been implemented on the SPHERES 167-MHz Digital Signal Processor (TI 6701 DSP) with 16 MB RAM. ✓ Thruster/mass-property algorithms are embedded C code, optimized to fit into 256 KB flash memory.
	Integrates on EDS platform		✗ No.
	Data I/O compatibility		✓ All algorithm results are transmitted to a laptop at 18 Kbps, as SPHERES currently has no non-volatile data storage. Thruster fault detection and mass-property algorithms have software interfaces (APIs) to support data I/O.
Maturity	TRL ≥ 6	4	✓ TRL is at least 6. Launched on STS-121 and flight tested on ISS.
	Reliability		✓ Technology verified in space.
	Stability		✓ Technology verified in space.

	Deployment history		✓ The thruster fault detection and mass-property algorithms have been implemented onboard the MIT SPHERES spacecraft. Experiments with SPHERES are being conducted on ISS and STS-121. ✓ The thruster fault detection and mass-property algorithms were tested in simulation on the JSC Mini-AERCam spacecraft. ✓ Extended testing in simulation of thruster fault detection and mass-property ID on the X-38, precursor to the Crew Return Vehicle.
Scalability	Proven in large-scale system	3	✗ Small spacecraft applications thus far.
	Scale-up		✓ The thruster algorithm is computationally efficient and scales better than linearly with the number of failure modes to be identified. The mass-property algorithm is computationally efficient and practical for real time.
	Distributed system		✓ Thruster fault detection and mass-property monitor distributed attitude control systems that have many subsystems and components.
Testability	Full test coverage	4	✓ 100% coverage of algorithms is attainable with a fixed number of test cases.
	Traceability		✓ Traceability of algorithm design to X-38 and Mini-AERCam fault detection requirements exists.
	Repeatable tests		✓ Deterministic algorithms give repeatable results with the same data.
	Number of test cases		✓ The number of test cases will be known from thruster configuration, directional failures, and possible combinations of failures.
Usability	Reusability	4	✓ Algorithms are reusable for similar systems and have been deployed on several platforms.
Supportability	Maintainability	4	~ Maintenance requires V&V of modifications to code and parameters, at least. ~ No specific support for maintenance, e.g., reusable model-based algorithm.
	Upgradability		~ Upgrade requires full V&V of algorithm on testbed.
	Quality of supporting organization		✓ These algorithms are developed by NASA Ames.
Cost	Relative cost	3	✓ No software license required. ✓ Development of code to flight standards.
	Highest cost risk		! Effort for integration and V&V testing on high-fidelity testbed/simulator. ! Cost of testbed/simulator/flight tests for V&V of adaptive control loop.
Schedule	Relative schedule	3	! V&V of adaptive control loop for a realistic system is hard.
	Highest schedule risk		! Access to high-fidelity testbed/simulator/flight tests.
Extensibility	Fleet operations support	2	✓ Algorithms can be applied across a fleet of vehicles, with preventative maintenance to fleet when faults are detected in one vehicle.
	Evolvable		✓ Attitude control fault detection and mass-property ID are needed by several Cx systems. Algorithms can be extended to similar RCS/TVC systems.

Table 7.2: Evaluation of TFPGs

FOM	TPM	Weight	Description
Coverage	False Negative Rate = 0	5	✓ The fault coverage is determined by what is modeled in the graph. 100% coverage of required faults and no missed fault detections are achieved, if all faults are properly modeled in the graph with required alarms for detection.
	Maximum Breadth Coverage		✓ There is no restriction on the types of faults that can be modeled, as long as there is a way of sensing an alarm that indicates the fault.
	Maximum Depth Coverage		✓ Failure modes can be modeled for components and a hierarchical graph of components constructed, to model how failure effects cascade from a low-level fault up the hierarchy and cause system-level effects.
Performance	Diagnostic Latency	5	? In a large scenario, 86 alarms were processed in ~9 seconds or 0.1 seconds per alarm on a 400-MHz MILSPEC PowerPC. ✓ The algorithm is bounded by fixing search parameters, to guarantee required response times for a modeled fault. It is not "anytime."
Accuracy	False Positive Rate = 0	5	✓ Diagnostic accuracy is determined by the quality and accuracy of the graph and alarms. If system and sensor failures are adequately modeled and there is sufficient sensor redundancy, the algorithm will cause no false alarms.
	Ambiguity		✓ The degree of ambiguity in the diagnosis is determined by the alarms in the system. Diagnosability analysis algorithms were developed to determine where additional sensing is needed to reduce ambiguity group size.
	Probability of diagnosis = 1.0		✓ The confidence in the diagnosis, which is computed by the algorithm, is determined by the graph and the currently active alarms.
Integrability	Integrates on VMS platform	4	✓ Runs on the VxWorks real-time operating system and PowerPC processor, using the Green Hills cross-compiler. ✓ TFPG algorithm is C++ code, ~10K SLOC. Memory usage depends on the size of the graph. Dynamic memory allocation and usage of the Standard Template Library should be removed for flight.
	Integrates on EDS platform		✗ Not suited for programmable logic chips, since algorithm runs a graph search.
	Data I/O compatibility		✓ Data input and results output are supported by a documented API. ✓ The graph is loaded on start-up from permanent storage in the embedded system, or fast-loaded from C++ object code in memory.
Maturity	TRL ≥ 6	4	✗ Technology Readiness Level is 4–5, tested in lab and flight experiment.
	Deployment history		✓ Boeing has had several projects involving TFPG/FACT.
	Reliability		? Limited experience with this tool.
	Stability		? Limited experience with this tool.

Scalability	Proven in large-scale system	3	✓ 215 fault scenarios were executed on a graph with 481 failure modes, 271 alarms, 1973 discrepancies, and 153 physical components (Boeing).
	Scale-up		✓ Algorithm complexity is polynomial, O (*n*+*m*) for *n* nodes and *m* edges in the graph. An upper bound on diagnostic latency can be computed for a graph. ! Use of the AND-connector graph construct should be avoided, as all connecting paths must be evaluated in the worst case.
	Distributed system		✗ No support provided.
Testability	Full test coverage	4	✓ 100% test coverage of the code is achievable. 100% test coverage of the graph plus algorithm is achievable. ✓ A simulation engine tests diagnosability, given an alarm allocation.
	Traceability		✓ The TFPG may be derived from fault trees and FMECAs so that faults modeled in the graph will be directly traceable to these documents.
	Repeatable tests		✓ The TFPG algorithm is deterministic. Search parameters are fixed during V&V testing to produce identical results in the same response time.
	Number of test cases		✓ The number of test cases for the code is fixed. Each fault propagation path through the graph and combination of alarms at various time intervals is verified by a separate test case.
Usability	Reusability	4	✓ The algorithm code is reusable; only the fault propagation graph is developed for the system to be diagnosed. ✓ The graph used in design for fault propagation and timing analysis can be deployed for on-board fault detection.
Supportability	Maintainability	4	✓ The graph may be modified without altering the algorithm source code, requiring retesting of only the graph with the algorithm.
	Upgradability		✓ A new graph and upgrades to the algorithm code require testing of the algorithm, as well as testing of the graph with the algorithm.
	Quality of supporting organization		✓ NASA could develop the software in house with diagnostic expertise from Ames and Vanderbilt ISIS.
Cost	Relative cost	3	✓ Low/no cost for TFPG software acquisition. ✓ Low cost for graph development.
	Highest cost risk		! Labor for design of diagnostic models. ! Development of the algorithm code to flight standards.
Schedule	Relative schedule	3	✓ Development of the graph for abort fault detection may begin at design.
	Highest schedule risk		! Access to subsystem experts to guide design of the fault graph. ! Development of the C++ algorithm code to flight standards.
Extensibility	Fleet operations support	2	✓ Faults detected using the TFPG can be used in fleet maintenance scheduling.
	Evolvable		✓ A new graph may be developed for CaLV/CEV, reusing the algorithm code.

Table 7.3: Evaluation of BEAM's DIAD and SIE

FOM	TPM	Weight	Description
Coverage	False Negative Rate = 0	5	✓ BEAM detects all fault modes that it has been trained on.
	Maximum Breadth Coverage		✓ BEAM has no upper limit on fault modes and operating modes that it can cover, given sufficient training data. ✓ Practical limit for a single, subsystem-level BEAM detector is ~ 30 modes and ~ 200 signals, due to training effort and access to training data.
	Maximum Depth Coverage		× DIAD and SIE are not hierarchical and do not have a range of fault isolation.
Performance	Diagnostic Latency	5	✓ SIE and DIAD have fixed-cost computation time per sensor sample. DIAD minimum latency is 1 sample; SIE minimum latency is roughly 10 samples. ✓ The latency of the detection is determined by the required confidence, and confidence improves with the number of samples. ✓ BEAM does not use search at any time.
Accuracy	False Positive Rate = 0	5	✓ DIAD trains on system failures and SIE on sensor failures, together preventing false alarms. The false alarm rate can be zero, given training data for known faults and assuming repeatable data. The false alarm rate is predictable at training time. ✓ Accuracy of the fault detection is limited not by the algorithm, but by the quality and quantity of its training. Training on fault data identifies significant faults; training on only nominal data (e.g., F/A-18 experiment) will detect all anomalies including previously unknown effects—possibly false positives. ✓ Diverse training data differentiates significant faults from nominal and other fault modes and deals with noise in the system/environment.
	Ambiguity		✓ Ambiguity can be eliminated by proper training to discriminate between fault modes, with different fault data sets. BEAM has no capability to suggest sensor placement to reduce ambiguity.
	Probability of diagnosis = 1.0		✓ Confidence grows with the number of samples observed over time. For a required confidence, the number of samples can be specified.
Integrability	Integrates on VMS platform	4	✓ BEAM has run on VxWorks on PowerPC 604. ✓ BEAM code is C++ and C; code size about 5K SLOC. ✓ Run-time memory requirements depend on the number of failure modes.
	Integrates on EDS platform		~ It is possible to run the algorithms on programmable logic chips (e.g., FPGA, DSP) but this has not been proven.
	Data I/O compatibility		✓ All BEAM modules have an API for data input and results output.
Maturity	TRL ≥ 6	4	✓ TRL is close to 6.

	Deployment history		✓ Cassini ✓ F/A-18 experiment, on a 300-MHz Geode 686.
	Reliability		~ False alarms can be caused by transients, noise, system instability, and quality of training data.
	Stability		~ Sensitivity of the data-driven approach to minor changes.
Scalability	Proven in large-scale system	3	✓ SIE was run on Cassini flight telemetry on 80 signals at 2 KHz and on over 950 signals at 8 Hz, on a Pentium desktop computer. ✓ In the F/A-18 experiment, SIE was run on 34 signals at 10 Hz and consumed less than 1% of CPU. All BEAM software used less than 2% CPU, most of which was verbose data I/O for test verification.
	Scale-up		✓ Complexity of the algorithm is O (*n*) for DIAD and O (n^2) for SIE, where *n* is the number of input signals. ✓ There is no run-time performance hit for additional failure modes. ✓ There is a cost in memory and training data for additional failure modes.
	Distributed system		✓ A distributed hierarchy using SHINE at the system level, with separate instances of DIAD and SIE for the subsystems, has been done. SHINE checks consistency of subsystem interactions using an overall system model.
Testability	Full test coverage	4	✓ 100% coverage of algorithm is attainable with a fixed number of test cases. ? Need to specify boundary/inclusive/exclusive test cases for data coverage.
	Traceability		~ Traceability from data features to fault detection requirements is not obvious. ~ Configuration management of data sets is required for each specific system.
	Repeatable tests		✓ Deterministic algorithm gives repeatable results with the same data.
	Number of test cases		? Number of test cases to verify different combinations of data is unknown.
Usability	Reusability	4	~ Data-driven approaches have limited reuse. The algorithm is reusable, however retraining may be required for even minor changes in the system.
Supportability	Maintainability	4	~ Sensitivity of the data-driven approach to minor changes.
	Upgradability		~ Sensitivity of the data-driven approach to major changes.
	Quality of supporting organization		✓ This algorithm is developed by NASA JPL.
Cost	Relative cost	3	~ Training cost is not amortized over successive deployments.
	Highest cost risk		! Effort to train on each system and configuration management of training data.
Schedule	Relative schedule	3	~ Training time is not amortized over successive deployments.
	Highest schedule risk		! Time to train for multiple systems and access to the systems.
Extensibility	Fleet operations support	2	~ Minor parameter variations will require retraining for similar vehicles, e.g., variations in hardware configurations, mission, and performance parameters.
	Evolvable		~ Data-driven algorithm requires training for new vehicles, e.g., CaLV.

Table 7.4: Evaluation of TEAMS-RT

FOM	TPM	Weight	Description
Coverage	False Negative Rate = 0	5	✓ As for other fault modeling approaches (e.g., TFPG), the model and tests are key to preventing missed detections. The model must be designed with coverage of all faults of interest, and tests must monitor and detect the faults. ✓ Testability analysis during design verifies that all fault modes are detectable.
	Maximum Breadth Coverage		✓ As for TFPG, the range of faults covered is driven by the tests. There is no restriction on the types of faults that can be modeled, as long as tests can be written to detect the fault.
	Maximum Depth Coverage		✓ As for TFPG, the hierarchical model supports fault isolation at multiple levels of abstraction, from system-level faults to component faults.
Performance	Diagnostic Latency	5	✓ TEAMS-RT has real-time performance in the millisecond range, excluding processing time of the test code that is dependant on the application's needs. ✓ TEAMS-RT reports diagnostic results almost immediately. The diagnosis is updated as test results of different latencies are received, i.e., anytime [20].
Accuracy	False Positive Rate = 0	5	✓ As for other fault modeling approaches (e.g., TFPG), the tests are key to preventing false alarms. Test code is external to the model and diagnostic algorithm and should not generate false positives. However, TEAMS-RT is resilient to inaccurate or uncertain tests allowing the use of more sensitive tests and more advanced statistical/signal processing code in the tests [23].
	Ambiguity		✓ At design time, testability analysis suggests additional sensing to better isolate faults in ambiguity groups. ✓ TEAMS-RT reports ambiguity groups in the diagnosis.
	Probability of diagnosis = 1.0		✓ Probabilities of diagnoses can be reported, based on known component failure rates (e.g., mean time between failure (MTBF)) or Bayesian combination of fault probabilities from the model.
Integrability	Integrates on VMS platform	4	✓ TEAMS-RT runs on embedded systems including VxWorks/PowerPC. ✓ TEAMS-RT is about 10K SLOC of C code.
	Integrates on EDS platform		~ D-matrix could be run on an FPGA but diagnostic engine cannot.
	Data I/O compatibility		✓ C code API supports integrating with other applications. ✓ TEAMS-RT accepts inputs from tests and outputs Good, Suspect, and Bad components.
Maturity	TRL ≥ 6	4	✓ COTS product with many industrial applications. ~ Has not flown in space yet.

	Deployment history		✓ X-33 RLV with NASA Ames and Rockwell. Real-time root-cause isolation for two 1000x1000 systems in 200 ms on a 40-MHz processor. ✓ Helicopter HUMS: engine monitoring of UH60/SH60 with Sikorsky, Goodrich, Army & Navy; T700 engine diagnostics for BlackHawk and Apache helicopters with Army and Boeing, GE, Honeywell. ✓ ISS telediagnosis with Honeywell and NASA JSC.
	Reliability		✓ Reliable product with proven track record in industry.
	Stability		✓ Stable product with decade-long track record.
Scalability	Proven in large-scale system	3	✓ Largest system is for Joint Strike Fighter.
	Scale-up		✓ Complexity is O (*n* ln *n*) where *n* is the number of nodes in the model.
	Distributed system		✓ TEAM-RT supports a distributed lattice architecture [22].
Testability	Full test coverage	4	✓ TEAMS-RT D-matrix and diagnostic engine can be 100% verified. ✓ Advanced support for testability analysis and testability metrics.
	Traceability		✓ Annotations in the model can be used to trace design to requirements; dependency tree traces fault modes to tests.
	Repeatable tests		✓ Deterministic algorithm gives repeatable results for real-time systems.
	Number of test cases		✓ Compact TEAMS-RT D-matrix and diagnostic engine (10K SLOC) and binary test inputs require limited number of test cases for full test coverage.
Usability	Reusability	4	✓ Diagnostic algorithms are reusable, only the model changes for different applications. ✓ Integrated TEAMS toolset provides continuity for the user from design through deployment and maintenance. ~ Primarily intended for fault modeling and monitoring; monitoring of nominal operations modes is not the focus of the toolset.
Supportability	Maintainability	4	✓ Direct support for maintenance by TEAMS-RDS, TEAMATE.
	Upgradability		✓ Model-based approach supports upgrades without impacting code as diagnostic algorithms are reusable.
	Quality of supporting organization		✓ Excellent. QSI has won several contracts based on performance, e.g., JSF.
Cost	Relative cost	3	✓ Licensing cost is $25K for TEAMS-RT/TEAMS-RDS.
	Highest cost risk		! Labor for design of diagnostic models. ! Code development to flight standards.
Schedule	Relative schedule	3	✓ Diagnostic models can be developed fairly quickly once design is known.
	Highest schedule risk		! Access to subsystem experts to guide design of the diagnostic models.
Extensibility	Fleet operations support	2	✓ Excellent life-cycle support for fleet of vehicles provided by TEAMS toolset.
	Evolvable		✓ Model-based approach supports growth of the diagnostic model to Block II.

5 Reference Documents

1. Honeywell Space Systems Glendale, "IVHM Analysis and Optimization," HSS-04 Task Report for Space Launch Initiative, 2003.
2. NASA ARC, "TA-5 Risk Reduction Integrated Vehicle Health Management State of the Art," 2nd Generation Reusable Launch Vehicle Program, 2002.
3. Edward Wilson, Chris Lages, and Robert Mah, "https://nx.arc.nasa.gov/nx/dsweb/Get/Document-114273/Thruster_FDI %282%29.pdfGyro-based maximum-likelihood thruster fault detection and identification," in *Proceedings of the 2002 American Control Conference*, Anchorage, Alaska, 2002.
4. Edward Wilson, Chris Lages, and Robert Mah, "On-line, gyro-based, mass-property identification for thruster-controlled spacecraft using recursive least squares," in *Proceedings of the 45th IEEE International Midwest Symposium on Circuits and Systems*, Tulsa, Oklahoma, 2002.
5. Edward Wilson, Dave Sutter, et al, "Motion-based system identification and fault detection and isolation technologies for thruster controlled spacecraft," in *Proceedings of the JANNAF 3rd Modeling and Simulation Joint Subcommittee Meeting*, Colorado Springs, CO, 2003.
6. NASA Headquarters, "Fault Tree Handbook with Aerospace Applications," Version 1.1, NASA Office of Safety and Mission Assurance, 2002.
7. S. Abdelwahed, G. Karsai, G. Biswas, "System Diagnosis using Hybrid Failure Propagation Graphs", Technical Report, ISIS-02-302, Institute for Software Integrated Systems, Vanderbilt University.
8. Larry Howard, "An Algorithm for Diagnostic Reasoning Using TFPG Models in Embedded Real-Time Applications". Proceedings of AUTOTESTCON IEEE Systems Readiness Technology Conference, 2001.
9. Sherif Abdelwahed, Gabor Karsai, and Gautam Biswas, "A Consistency-based Robust Diagnosis Approach for Temporal Causal Systems," 16th International Workshop on Principles of Diagnosis (DX 05), Monterey, CA, 2005.
10. A. Misra, J. Sztipanovits, J.R. Carnes, "Robust diagnostic system: structural redundancy approach", 1994.
11. Atlas, L.; Bloor, G.; Brotherton, T.; Howard, L.; Jaw, L.; Kacprzynski, G.; Karsai, G.; Mackey, R.; Mesick, J.; Reuter, R.; Roemer, M., "An evolvable tri-reasoner IVHM system," Proceedings of IEEE Aerospace Conference, 2001.
12. A. Misra, J. Sztipanovits, A. Underbrink, R. Carnes, B. Purves, "Diagnosability of Dynamical Systems," Third International Workshop on Principles of Diagnosis, Rosario, WA, 1992.
13. R. Mackey, M. James, H. Park, and M. Zak, "Beacon-Based Exception Analysis for Multimissions: Technology for Autonomous Self-Analysis," NASA JPL TMO Progress Report 42-144, 2001.
14. Michail Zak, Han Park, "Grey-box Approach for Fault Detection of Dynamical Systems," Proceedings of IEEE Aerospace Conference, 2001.
15. Park, H. G., Mackey, R., James, M., Zak, M., Kynard, M., Greene, W., Sebghati, J., "Analysis of Space Shuttle Main Engine Data Using Beacon-based Exception Analysis for Multimissions," IEEE Aerospace Conference, 2002.

16. S. Gulati, R. Mackey, "BEAM: Autonomous Diagnostics and Prognostics for Complex Spaceborne Systems," poster session, 1998 AIAA Defense & Civil Space Programs Conference and Exhibit, Huntsville, AL, 1998.
17. Ryan Mackey, David Iverson, Greg Pisanich, Mike Toberman, Ken Hicks, "ISHM Technology Demonstration Project Final Report", NASA TM 2006-213482, 2006.
18. S. Gulati, R. Mackey, M. James, H. Park, "VHM Technology Validation: RLV/X-33 Analysis," poster session, 2nd International Workshop on Structural Health Monitoring, Palo Alto CA, 1999.
19. Expert Microsystems, Inc, "MSET Signal Validation System Final Report," on the SureSense™ Signal Validation System for NASA Contract NAS8-98027, 2000.
20. Somnath Deb, "Real-time Onboard and remote Vehicle Health Management", Phase II STTR Final Technical Report on NAS2-02061(CDT), 2004.
21. Deb S., Pattipati K.R., Raghavan V., Shakeri M., Shrestha R., "https://nx.arc.nasa.gov/nx/dsweb/Get/Document-140523/autotestcon94.pdfMulti-Signal Flow Graphs: A Novel Approach for System Testability Analysis and Fault Diagnosis". Proceedings of AUTOTESTCON '94 IEEE Systems Readiness Technology Conference, 1994.
22. S. Deb, A. Mathur, P.K. Willett, K.R. Pattipati, "https://nx.arc.nasa.gov/nx/dsweb/Get/Document-159940/smc_rt_98.pdfDe-centralized Real-time Monitoring and Diagnosis". Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, 1998.
23. Somnath Deb, Venkata N. Malepati, Michel D. Paquet, Baban Baliga, "Validation of a COTS EHM Solution for the JSF Program," IEEE Aerospace Conference, Big Sky, MT, 2006.
24. Mohammad Azam, Krishna Pattipati, Jeffrey Allanach, Scott Poll, Ann Patterson-Hine, "In-flight Fault Detection and Isolation in Aircraft Flight Control Systems", IEEE Aero Conference, 2005.
25. M. Holthaus, "Model Documentation for CLIN 0001, Engineering Support Under Qualtech's NASA Contract Entitled "Multisignal Flow Graphs for System Fault Diagnosis," Rockwell Aerospace, Final Report, 1996.
26. Mohammad Azam, David Pham, Fang Tu, Krishna Pattipati, Ann Patterson-Hine, and Lui Wang, "Fault Detection and Isolation in the Non-Toxic Orbital Maneuvering System and the Reaction Control System," Proceedings of the IEEE Aerospace Conference, Big Sky, MT, 2004.
27. Somnath Deb, Charles Domagala, Ghoshal S., Patterson-Hine A., Alena R., "Remote Diagnosis of the International Space Station utilizing Telemetry Data". Proceedings of SPIE, Component and Systems Diagnostics, Prognosis, and Health Management, 2001.

Appendix A – Apollo EDS Abort Decision Checkpoints

Table 8: Saturn V Abort Limits

Parameter	Abort Limits
Attitude	Off attitude for 0.6 seconds
Attitude Rate	± 4 to 20 deg/sec
Q-alpha	100% and high rate
Engine failure	Function of stage, # failures, and rates
Separation	Failure to separate
Tank Pressure	Varied by tank

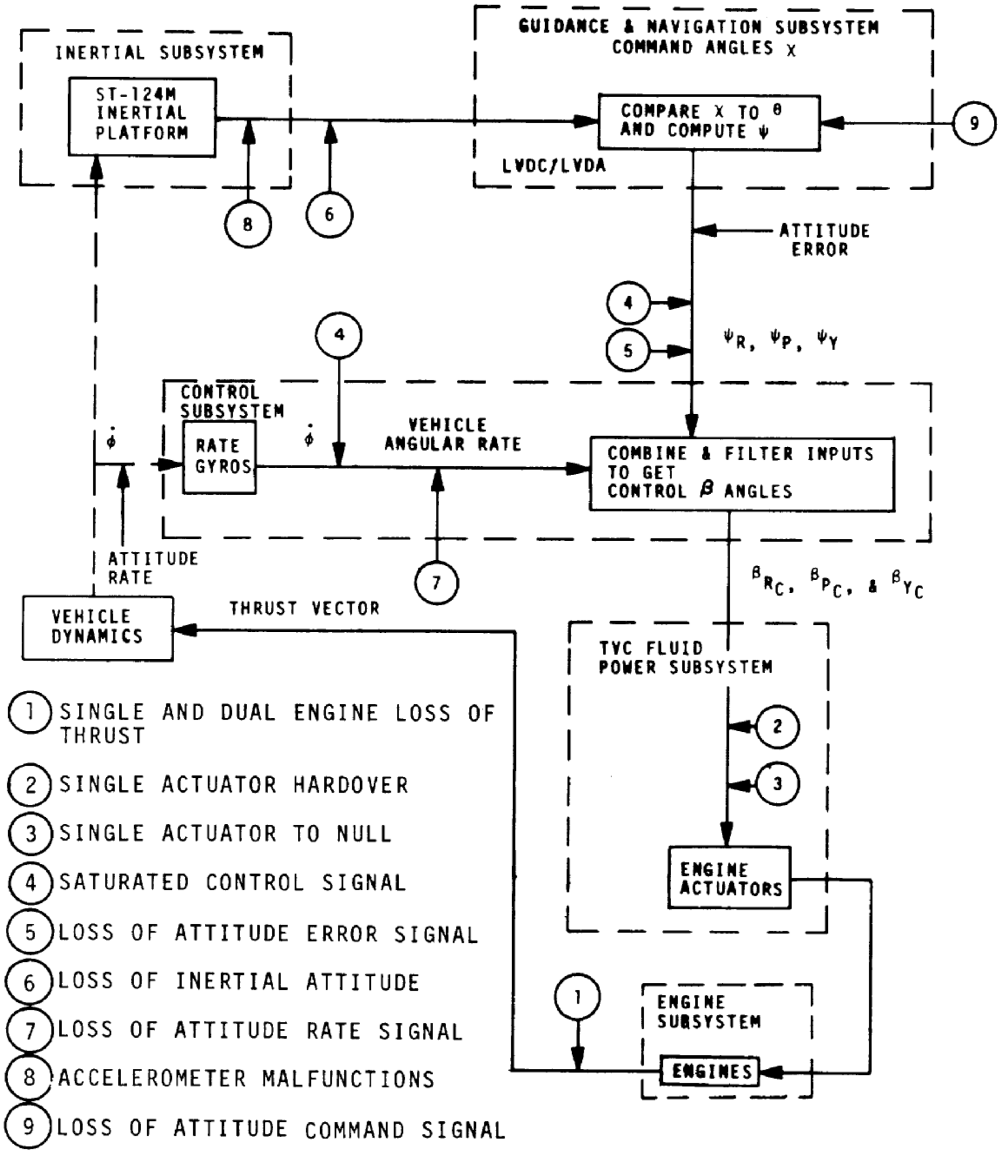
INERTIAL SUBSYSTEM
ST-124M INERTIAL PLATFORM
GUIDANCE & NAVIGATION SUBSYSTEM
COMMAND ANGLES χ
COMPARE χ TO θ AND COMPUTE ψ
LVDC/LVDA
ATTITUDE ERROR
ψR, ψP, ψY
CONTROL SUBSYSTEM
RATE GYROS
φ̇
VEHICLE ANGULAR RATE
COMBINE & FILTER INPUTS TO GET CONTROL β ANGLES
ATTITUDE RATE
βRC, βPC, & βYC
VEHICLE DYNAMICS
THRUST VECTOR
TVC FLUID POWER SUBSYSTEM
ENGINE ACTUATORS
ENGINE SUBSYSTEM
ENGINES
1
2
3
4
5
6
7
8
9
1 SINGLE AND DUAL ENGINE LOSS OF THRUST
2 SINGLE ACTUATOR HARDOVER
3 SINGLE ACTUATOR TO NULL
4 SATURATED CONTROL SIGNAL
5 LOSS OF ATTITUDE ERROR SIGNAL
6 LOSS OF INERTIAL ATTITUDE
7 LOSS OF ATTITUDE RATE SIGNAL
8 ACCELEROMETER MALFUNCTIONS
9 LOSS OF ATTITUDE COMMAND SIGNAL

Figure 2: Usage of Saturn V Abort Limits

Appendix B – Assessment Criteria for Ground and Design

Table 9: Assessment Criteria for Ground Operations Fault Detection

FOM	TPM	Weight	Description
Coverage	False Negative Rate = 0	5	100% coverage of required faults. Zero missed abort faults.
	Maximum Breadth Coverage		Range of fault classes covered.
	Maximum Depth Coverage		Multiple levels of abstraction covered.
	Prognosis		Supports trending and prediction of faults, and determines remaining life.
Performance	Diagnostic Latency	5	Latency < Time-To-Criticality minus margin for crew escape. Maximize time for crew escape, minimize detection delay. Guarantees diagnosis in hard/soft real-time.
Accuracy	False Positive Rate = 0	5	No false alarms which trigger unnecessary crew abort or halt the countdown. No false alarms for critical sensors. This criteria is not as firm as for on-board—some false positives are tolerated, to catch anything unusual before launch.
	Ambiguity		Isolates to root cause fault, exonerating components with dependent symptoms.
	Probability of diagnosis = 1.0		Confidence in the abort recommendation, resolution of ambiguity.
Integrability	Integrates on Launch Management System (LMS) platform	4	Compatibility with LMS operating system (e.g., Unix) and computing environment (e.g., Sun workstation). Diagnostic application encoded in modern software language (e.g., C/C++). Code size and memory are not restricted as for the flight system.
	Integrates on EDS platform		Compatibility with programmable logic chips, e.g., FPGA.
	Data I/O compatibility		Supports integration via API to LMS data acquisition, for CLV and ground support equipment (GSE) sensor data and telemetry; and output of diagnostic results, e.g., LMS displays and data repositories.
Maturity	TRL ≥ 6	4	Technology Readiness Level of the current technology.
	Deployment history		Number of relevant deployments in NASA, aerospace industry.
	Reliability		Robust and fail-proof, based on bug-tracking history and user problem reports.
	Stability		The current version of the technology meets its intended purpose.
Scalability	Proven in large-scale system	3	Relevant deployments in NASA, aerospace industry.
	Scale-up		Supports fault coverage for First Stage, Upper Stage, and Upper Stage Engine.
	Distributed system		Supports fusion of results from First Stage, Upper Stage, and Upper Stage Engine diagnostic systems.
Testability	Full test coverage	4	Conventional verification for all required fault scenarios. 100% flight code coverage. Specialized V&V tools meet reliability and certification requirements.

	Traceability		Requirements traceable through DDT&E artifacts.
	Repeatable tests		Deterministic diagnostic software/V&V tools. Repeatable results are obtained within real-time requirements.
	Number of test cases		Verification of expected results for all input data used by the technology can be verified with a feasible number of test cases, e.g., inputs are discrete and sparse, or continuous and the range can be broken up into discrete sets covering special/ boundary cases and exceptions.
Usability	Situation assessment	3	Supports user displays and reports for rapid, accurate situation assessment. Diagnostic information supports identified tasks and workflow. Presentation is concise/intuitive and relevant to operations context. User interfaces support training.
	Collaborative environment		Effective presentation of diagnostic information to multiple users, relevant to their tasks, e.g., mission manager, technician, etc. Supports communication for cooperative tasks/workflow. Interactive troubleshooting accepts technician input to reduce diagnostic ambiguity.
	On-line documents and data archives		Supports access to on-line documents for supporting diagnostic evidence and explanations for repair. Searchable operations manuals and maintenance procedures, with automatic reference lookup based on the fault. Support for searchable repositories of historical mission data.
	Reusability		Diagnostic system design factors out common reusable parts, e.g., object-oriented software with code reuse, model-based diagnosis with reuse of the diagnostic engine, and reuse of generic component models.
Supportability	Maintainability	4	Effort to maintain the diagnostic system once operational. Availability of manuals and user guides.
	Upgradability		Effort to upgrade the diagnostic system once operational. Sensitivity to change, based on deployment history or degree of partitioning/reuse in the design.
	Quality of supporting organization		Support for DDT&E, maintenance, upgrade, and training. Stability and CMMI level of the organization. Availability of knowledgeable experts throughout the life cycle.
Cost	Relative cost	3	Relative cost for this technology, e.g., labor for DDT&E, licensing, materials
	Highest cost risk		Likely highest cost risk for this technology.
Schedule	Relative schedule	3	Relative schedule for this technology, e.g., time for DDT&E and acquisition.
	Highest schedule risk		Likely highest schedule risk for this technology.
Extensibility	Fleet operations support	2	Use of fault information across the fleet for logistics supply and maintenance. Use of historical data for prognostics and preventative maintenance.
	Evolvable		Applicability to Exploration Systems, e.g., CaLV and Lunar missions.

Table 10: Assessment Criteria for Design Tools for Fault Detection

FOM	TPM	Weight	Description
Coverage	Testability analysis	5	100% coverage of abort faults and required non-critical faults in the diagnostic model for CLV instrumentation and fault detection capability.
	Maximum Breadth Coverage		Capability to represent and analyze a wide range of fault classes and domains. A broad modeling approach supports a diverse set of failure scenarios, e.g., empirical modeling techniques for capturing knowledge about statistical phenomena as well as discrete modeling of nominal and anomalous behavioral modes.
	Maximum Depth Coverage		The depth of the diagnostic modeling approach refers to the capability to model multiple levels of abstraction, as in a hierarchical structure from low-level devices (e.g., sensors in a pump system) to high-level systems (e.g., subsystem assembly or vehicle).
	Fault analysis tools		Support for Fault Trees, FMECA, PRA, Functional Modeling. Analysis using criticality and reliability metrics, e.g., failure rate, MTBF. Ambiguity group sizes and probability distributions. Analysis tools support comparison of alternate fault models to optimize coverage and sensor placement.
Performance	Sensor sampling rates	3	Support for determining sensor sampling rates required to maximize time for crew escape.
	Design tool performance		Adequate tool performance in networked environment with multiple users.
Accuracy	Sensor placement optimization	5	Sensor placement that minimizes ambiguity by isolating faults.
Integrability	Runs in conventional computing environments	3	Compatible with desktop computing platforms. Uses modern programming language, e.g., C/C++. Supports storage of diagnostic design and models on networked data repository for shared use. Design tool API allows integration with external tools, e.g., V&V tools.
Maturity	User community	4	Relevant users in NASA, aerospace/industry.
	Reliability		Robust and fail-proof tool with a history of stable software releases.
Scalability	Proven in large-scale system	3	Relevant deployments in NASA, aerospace industry.
	Scale-up		Supports fault models for both First Stage and Upper Stage.
Testability	V&V of fault models	4	Support for maintenance of design consistency throughout life cycle, i.e., traceability from requirements to design; from design to implementation. Supports annotation with rationale for design decisions. Availability of specialized tools for V&V of the diagnostic models developed with this tool.

Usability	Knowledge capture and modeling	4	Support for capturing diagnostic knowledge from domain information, data or experts and representing it in a diagnostic model or similar form that facilitates analysis. Efficiency and effectiveness of the knowledge capture and modeling processes. Quality and ease of use of the design tool, including how intuitive modeling is and whether the capture process is well supported by the interface. Graphical display of models and generation of reports for effective presentation of fault analysis information.
	Collaborative environment		Supports multiple designers with configuration management and version control.
	Reusability		Reusability of the design, e.g., reuse of diagnostic models for common components and object-oriented software designs.
Supportability	Maintainability	4	Effort to maintain the diagnostic design, using this tool.
	Upgradability		Effort to revise the diagnostic design for CLV upgrades, using this tool.
	Quality of supporting organization		Training in the use of the design tool and support for design, maintenance, and upgrades. Stability of the vendor and availability of support throughout the CLV life cycle.
Cost	Relative cost	3	Relative cost for this technology, e.g., labor for design, licensing.
	Highest cost risk		Likely highest cost risk for this technology.
Schedule	Relative schedule	3	Relative schedule for this technology, e.g., time for fault modeling.
	Highest schedule risk		Likely highest schedule risk for this technology, e.g., knowledge acquisition
Extensibility	Fleet operations support	2	Use of fault and performance data from the fleet for design upgrades.
	Evolvable		Design tool is compatible with commonly available IDE(s) for add-on extensions (e.g., V&V tools) and design tool enhancements (e.g., auto-generation of fault protection software from an integrated diagnostic modeling environment).

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 9/6/2006	**2. REPORT TYPE** NASA STI Technical Memorandum	**3. DATES COVERED** *(From - To)* February 2006-June 2006

4. TITLE AND SUBTITLE

DIAGNOSTIC TECHNOLOGY EVALUATION
REPORT FOR ON-BOARD CREW LAUNCH
VEHICLE

5a. CONTRACT NUMBER
NNA04AA18B

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
WBS: 136905.08.05.08.01.02.01 (CLV US ISHM)

6. AUTHOR(S)

Sandra Hayden	Ryan Mackey	Scott Poll
Nikunj Oza	Sriram Narasimhan	Somnath Deb
Robert Mah	Gabor Karsai	Mark Shirley

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NASA Ames Research Center
Moffett Field, CA 94035-1000

Institute for Software-Integrated Systems
Vanderbilt University
Box 1829, Station B
Nashville, TN 37235

NASA Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109

Qualtech Systems, Inc.
Putnam Park, Suite 603, 100 Great Meadow Road
Wethersfield, CN 06109

8. PERFORMING ORGANIZATION REPORT NUMBER
NASA/TM-2006-214552

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITOR'S ACRONYM(S)
NASA

11. SPONSORING/MONITORING REPORT NUMBER
NASA/TM-2006-214552

12. DISTRIBUTION/AVAILABILITY STATEMENT

NASA Personnel and NASA Contractors Only

13. SUPPLEMENTARY NOTES

Point of Contact: Sandra Hayden, NASA Ames Research Center, MS 269-3,
Moffett Field, CA 94035-1000 (650) 604-1676

14. ABSTRACT

In the decades since Apollo and the development of Space Shuttle, many new and diverse diagnostic technologies have been developed that present opportunities for improved fault detection and diagnosis. This report evaluates current diagnostic technologies for fault detection and diagnosis on-board CLV. The goal of the report is not to recommend technologies to be hosted on-board CLV; rather, it is to identify what is state of the practice that could realistically be flown and what is leading-edge state of the art that cannot be ready for flight, then to use this knowledge to assist the development of feasible requirements for CLV fault detection and diagnosis.

15. SUBJECT TERMS

diagnosis, algorithms, Crew Launch Vehicle, fault detection, diagnosis and recovery (FDDR), thruster fault detection, real-time mass property estimation, fault propagation, anomaly detection, sensor validation, model-based diagnosis, figures of merit, technical performance measures

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** ***(Include area code)***
U	U	U	UU	48	